Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia
Departamento de Informática

Dissertação de Mestrado

Mestrado em Engenharia Informática

Avaliação de condições de fiabilidade e segurança de protocolos de encaminhamento de dados em redes de sensores sem fios

Pedro Miguel Oliveira Marques da Silva (nº 26649)

1º Semestre de 2009/10

5 de Fevereiro de 2010

Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia
Departamento de Informática

Dissertação de Mestrado

Avaliação de condições de fiabilidade e segurança de protocolos de encaminhamento de dados em redes de sensores sem fios

Pedro Miguel Oliveira Marques da Silva (nº 26649)

Orientador: Prof. Doutor Henrique João Lopes Domingos

Trabalho apresentado no âmbito do Mestrado em Engenharia Informática, como requisito parcial para obtenção do grau de Mestre em Engenharia Informática.

1º Semestre de 2009/10

5 de Fevereiro de 2010

Agradecimentos

Nº do aluno: 26649

Nome: Pedro Miguel Oliveira Marques da Silva

Título da dissertação:
Avaliação de condições de fiabilidade e segurança de protocolos de encaminhamento de dados em redes de sensores sem fios

Palavras-Chave:

- Redes de sensores sem fios
- Protocolos de encaminhamento seguros
- Simulação de redes de sensores
- Ataques por intrusão

Keywords:

- Wireless Sensor Networks
- Secure Routing Protocols
- WSN Simulation
- Intrusion Attacks

Resumo

As redes de sensores sem fios (RSSF) são uma tecnologia emergente e de grande interesse para diferentes aplicações. São constituídas por pequenos dispositivos que cooperam entre si e se auto-organizam, podendo formar uma rede de larga escala, com milhares de nós, que cobrem uma área geográfica de grande dimensão. O funcionamento autónomo dos dispositivos e a característica de auto-organização da rede, no seu conjunto, apresentam alguns desafios relacionados com a segurança, nomeadamente no que concerne à segurança dos dados processados e encaminhados ao longo da rede.

O trabalho a realizar pretende contribuir para o estudo das condições de segurança e robustez dos mecanismos e protocolos de encaminhamento para RSSF. Para o efeito pretende-se que este estudo se realize a partir de um modelo sistémico de avaliação e comparação experimental de protocolos, a partir de uma plataforma de simulação que permita avaliar a sua operação em condições próximas de condições de operação real. A plataforma de simulação será dotada das ferramentas e componentes necessários à avaliação experimental de protocolos de encaminhamento seguro com tolerância a intrusões. Estes protocolos conjugam mecanismos e contra-medidas de defesa contra ataques às comunicações, de tipologia inspirada no modelo de adversário de Dolev-Yao. Conjugam ainda, mecanismos de prevenção de ataques específicos ao encaminhamento, podendo estes ser desencadeados a partir da injecção ou da replicação de comportamentos incorrectos de operação, em nós que tenham sido objecto de ataques por intrusão.

Como contributo principal deste trabalho, destaca-se a concepção de um ambiente de simulação inovador, uma vez que se pretendem implementar funcionalidades não encontradas nos sistemas de simulação para as RSSF existentes. Este sistema possibilitará desenhar e avaliar algoritmos de encaminhamento, concebidos para serem seguros e que constituem referências na investigação recente. Esta avaliação estará centrada fundamentalmente na análise da eficácia dos mecanismos de segurança previstos e, igualmente, na medida e na avaliação do impacte desses mecanismos em relação ao consumo de energia, à fiabilidade, à latência e à resistência do protocolo face às condições de cobertura e escala da rede.

Abstract

Wireless Sensor Networks (WSN) are an emerging and very interesting technology applied to different applications. They are formed by small, self organized devices that cooperate to form a large scale network with thousands of nodes covering a large area. The independent operation of the devices and the self-organization feature of the network present some challenges related to security, particularly regarding the security of the processed and routed data over the network.

This work aims to contribute to the study of the security conditions and robustness criteria of routing protocols for WSN. This study must be performed from a systemic model of assessement and from an experimental comparison of protocols, based upon a simulation platform which will enable to assess their operation in closed-to-real operating conditions. The simulation platform will be equipped with the needed tools and components, in order to enable experimental evaluation of secure and intrusion tolerant protocol. These protocols combine mechanisms and defensive counter-measures against attacks on communications, inspired by the Dolev-Yao adversary model. They include also mechanisms to prevent specific routing attacks, which can be triggered from the injection or from the replication of misbehavior of operation, in nodes that have been subjected to intrusion attacks.

The major contribution of this work is the design of an innovative simulation environment, since some of its features are not found in existing WSN simulation systems. It will provide the opportunity to implement and evaluate routing algorithms that are designed to be secure but for which there are no experimental studies on the real robustness and impact of designed security mechanisms. This evaluation will focus primarily on examining the effectiveness of the provided security mechanisms. Additionally, it will also assess the impact of these mechanisms in relation to energy consumption, reliability, latency and resistance of the protocol, regarding the coverage and the scale of the network.

Conteúdo

Lista de Figuras

Lista de Tabelas

1 . Introdução

1.1 Introdução geral

Recentemente, têm-se observado avanços na concepção e fabrico de sistemas computacionais programáveis, baseados em *hardware* de pequena dimensão, [69] com capacidade para desempenhar tarefas específicas. Estes avanços permitiram integrar, nesses sistemas, processadores miniaturizados, memória, dispositivos de processamento de sinal e de conversão analógica-digital para a detecção de diferentes fenómenos físicos (através de diversos tipos de sensores) e capacidades de comunicação sem fios por rádio frequência (com base nas normas 802.15.4 [17, 22] e Zigbee [22])). A possibilidade de construção destes dispositivos (que se designam mais simplesmente por nós sensores) fez surgir, nos últimos anos, um novo campo da investigação conhecido por redes de sensores sem fios (RSSF).

Uma RSSF é formada por um conjunto de dispositivos com as características descritas anteriormente, distribuídos numa certa área geográfica, que podem funcionar de forma autónoma ou sem supervisão humana. Estas redes permitem monitorizar, com maior ou menor densidade, diferentes fenómenos físicos associados ao meio ambiente envolvente. As RSSF possuem características de auto-organização, podendo ser formadas por um menor ou maior número de sensores, permitindo cobrir desde pequenas a vastas áreas de monitorização. Um ambiente de instalação de uma rede pode ser um edifício, uma instalação industrial, uma área de combate, uma vasta zona de monitorização de um habitat natural, um veículo ou o próprio corpo humano [67, 50, 51, 19].

A componente básica e fundamental de uma rede de sensores é, pois, o nó sensor (também designado por *mote*) [8, 1, 14, 9]. Cada um destes nós constitui um substrato computorizado que pode possuir diversos sensores para monitorizar, por exemplo, temperatura, luz, movimento e outros fenómenos físicos, consoante as necessidades das aplicações. Sendo um dispositivo miniaturizado e concebido para possuir um baixo custo de produção, apresenta um poder de computação limitado, baixa largura de banda de comunicação, curto alcance de comunicação rádio e energia autónoma limitada (com base em baterias do tipo AAA ou de células fotovoltaicas) [73]. Numa RSSF a energia pode constituir um recurso finito. Em certos ambientes de instalação pode não ser possível, ou viável, realizar operações que exijam intervenção ou supervisão humana, por exemplo, para o carregamento ou substituição de baterias. Conhecidas

estas limitações, para se poderem atingir distribuições de grande cobertura geográfica, os sensores têm de ser distribuídos em grande número, podendo-se também, por esse meio, aumentar a redundância dos nós que, assim, formam redes de larga escala que chegam a atingir milhares de nós.

Os *motes* podem diferir uns dos outros consoante a sua função na rede e poderão desempenhar, fundamentalmente, dois papeis: nó genérico gerador de informação (*source-nodes*) e nó base ou de sincronização (nó colector de dados da rede ou de execução de comandos de pesquisa). Numa RSSF os nós podem também actuar como nós de interligação (ou de encaminhamento e processamento intermédio através da rede) ou *gateways* (que permitem ligar o ambiente da rede de sensores a outras redes e sub-sistemas externos). Uma RSSF pode ser concebida de forma a ser interligada a outras infraestruturas computacionais que, com maior capacidade de armazenamento e processamento, permitem efectuar análise de dados coligidos. Na tecnologia actual existem ainda nós de desenvolvimento, que possuem ligações a computadores (ex: ligação de rede *Ethernet*, RS-232 ou USB), permitindo o carregamento expedito de código desenvolvido em estações de desenvolvimento. Os sensores dotados de ligações Ethernet, RS-232 ou USB podem ainda funcionar como nós de tipo *gateway*, permitindo, num cenário concreto, ligar a rede de sensores a aplicações, executando em sistemas de computadores usuais [3].

1.1.1 Caracterização de RSSF

As redes de sensores sem fios podem ser abordadas como um caso especial de redes *ad-hoc*, embora exibam características específicas [25]. As RSSF para aplicações de larga escala fazem emergir algumas problemáticas inerentes a aplicações distribuídas, com especificidades e desafios próprios [26], nomeadamente ao nível dos mecanismos de gestão, da organização topológica autónoma, das necessidades de sistemas de encaminhamento *multi-hop*, de requisitos de tolerância a falhas, de requisitos de escalabilidade ou de necessidade de serviços de segurança.

1.1.2 Aplicações

Muitas foram as aplicações [19, 45] encontradas na investigação ou na utilização emergente de RSSF com diferentes requisitos de escala [73]. O carácter autónomo destas redes oferece um sem número de vantagens que propicia a sua utilização em locais remotos de acessibilidade

difícil e onde não é possível a sua manutenção e supervisão. De entre as aplicações das RSSF, podem destacar-se as seguintes:

Detecção de alvos/objectos(*Target Tracking*): [67] associada à detecção de movimento (trajectória/presença) em áreas vigiadas (como, por exemplo, em teatros operacionais militares ou na vigilância e monitorização de recursos ou infraestruturas);

Monitorização de fenómenos naturais: [50] associada à detecção de eventos ou anomalias ambientais (com aplicações na agricultura, monitorização de poluição ou monitorização de habitats naturais), bem como de vigilância ou controlo de fenómenos naturais (sismos, actividade de vulcões, etc);

Recolha de dados: [51] associada ao controlo de indicadores físicos ou biomédicos de pessoas ou de animais (com recurso a sensores especiais, associados a aplicações da medicina) ou como ambientes de monitorização de operação de infraestruturas críticas (pontes, edifícios, sistemas electromecânicos ou equipamentos de instalações fabris).

1.2 Segurança em RSSF

A segurança nas RSSF é, *de facto*, um problema, quando se perspectiva a sua utilização para sistemas críticos. A segurança deve ser pensada em tempo de concepção [59], tendo em vista a abrangência do sistema e tendo em atenção as particularidades específicas da tecnologia inerente, bem como dos ambientes onde são implementadas. Importa analisar as hipóteses de desencadeamento de ataques a estas redes (a partir da definição de modelos de adversário) bem como as repercussões das potenciais tipologias de ataques que representem o modelo de adversário [75, 43]. Esta análise deve ser feita tendo em conta a pilha de protocolos [19] e de serviços associados ao *software* [15, 2, 44, 61] que executa em cada nó, uma vez que cada uma das camadas de serviços e protocolos pode ser vulnerável a esses ataques.

Na abordagem de uma plataforma usual e genérica para um nó de uma RSSF, verifica-se que, em geral, cada nó apresenta uma pilha minimalista de protocolos e serviços, por comparação com uma pilha associada a uma rede de computadores usual (ex., TCP/IP ou pilha OSI) [66]. As limitações impostas pelas dimensões e as capacidades de operação não permitem uma

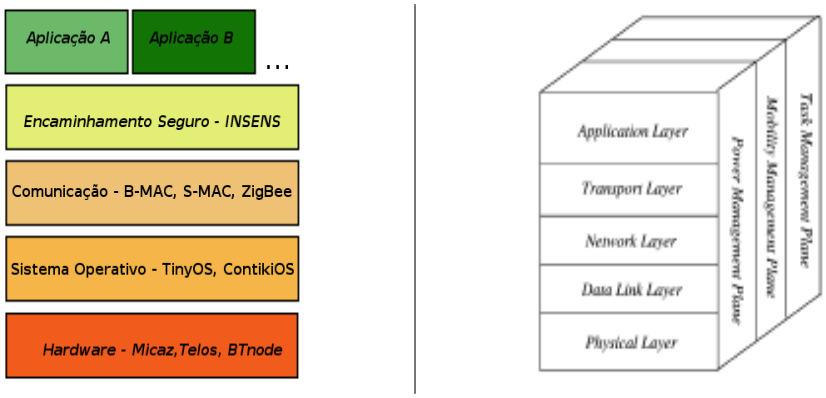

Figura 1.1 Pilha de serviços e pilha de protocolos [19] de uma RSSF

arquitectura muito ambiciosa e, por outro lado, as RSSF possuem geralmente uma vocação orientada para aplicações específicas, que condicionam os serviços que devem ser suportados na pilha. As camadas de operação de um nó sensor são essencialmente cinco [19]: camada física, camada de ligação de dados, camada de rede, camada de transporte e camada de aplicação. Todavia, na maior parte dos casos, a camada de transporte de dados e a funcionalidade inerente à camada de rede são concebidas de forma mais ou menos específica, tendo em vista as características particulares das aplicações. Na investigação, verifica-se ainda que a camada de ligação de dados (nível MAC e protocolos data-link) foram objecto de várias propostas, com diferentes variantes que podem apresentar vantagens particulares, dados os requisitos de operação das aplicações [62, 72, 27].

Alguns autores [70, 28, 57] têm vindo a desenhar algoritmos com vista a minimizar o impacte dos ataques ao encaminhamento, durante a operação das RSSF. Estes algoritmos pretendem garantir algumas propriedades básicas de segurança [64] (ex: confidencialidade, integridade, autenticação, detecção de retransmissão ilícita de dados). Não obstante, devem ser consideradas outras tipologias de ataques especificamente associados ao suporte de encaminhamento de dados na rede. Diferentes protocolos de encaminhamento seguro em RSSF endereçam apenas algumas dessas tipologias mas, em geral, não contemplam contra-medidas globais face a todas. Por outro lado, muitas das propostas apresentadas têm por base modelos matemáticos, análises teóricas ou experiências de pequena dimensão. Torna-se importante que esse estudo teórico seja complementado e verificado, tendo em conta o desempenho desses protocolos com análises experimentais mais próximas dos ambientes reais em que a rede opera. Para tal, o recurso a ambientes de simulação torna-se uma direcção importante de apoio a este estudo.

1.3 Objectivos e contribuições

Uma das vertentes do estudo da segurança em RSSF tem que ver com a possibilidade de se poderem efectivar ataques ao nível do encaminhamento de dados (da pilha de suporte de serviços de *software*, Figura 1.1). Diferentes tipologias de ataques [43, 37, 59] exigem diferentes tipos de contra-medidas. Estas, normalmente, são combinadas em mecanismos de segurança, consoante as propostas de sistemas de encaminhamento seguro. Estes são implementados tendo em conta as características e o modo de operação das RSSF [61, 47, 44].

Conhecidas que estão as dificuldades existentes no estudo de protocolos de encaminhamento seguro [43, 52], estes permanecem como um dos aspectos em aberto e como um desafio à concepção de RSSF que operem em condições de segurança. Este desafio é tanto mais relevante quando a análise de segurança pode envolver a avaliação de diferentes modelos e hipóteses de adversário [29, 56] e tipologias de ataques [75, 43]. Estas tipologias nem sempre são estudadas de forma sistemática e comparativa, na abordagem de diferentes protocolos de encaminhamento seguro que vão sendo propostos. Por outro lado, existe uma dificuldade adicional em poder conjugar-se o estudo das contra-medidas de segurança importantes com a sua avaliação experimental face à implementação. Para cada protocolo interessa medir o impacte que diferentes tipologias de ataques podem ter, nomeadamente, em cenários de grande escala, o que exige a utilização de sistemas de simulação [11, 5, 12, 13, 6, 4] de RSSF que permitam simular diferentes hipóteses de ataque e antecipar o seu impacte. Este impacte deve ser analisado não apenas no que refere ao comportamento dos protocolos mas, complementarmente, no que refere à forma como afectam a própria rede nomeadamente, em termos de consumo de energia, fiabilidade e latência. Um sistema que suprima estas dificuldades contribui para um mais rápido desenvolvimento e uma afinação mais cuidada de determinados parâmetros dos protocolos, com vista a garantir as propriedades de segurança desejadas para o ambiente de operação das RSSF, em condições mais realistas.

No âmbito deste trabalho foi concebido e desenvolvido um sistema de simulação inovador, uma vez que conjuga a facilidade de desenhar/configurar a topologia da rede (com recurso a um ambiente gráfico), com a existência de ferramentas generalistas para análise do comportamento das RSSF, com ou sem ataques dirigidos à rede. Assim, este sistema permite o estudo sistemático de protocolos de encaminhamento, quer sejam desenhados ou não para serem seguros, e possui, em particular, as seguintes funcionalidades:

- Interface de visualização e configuração da rede com informações dos parâmetros de

simulação e informação de cada nó (por exemplo, o seu estado energético, identificador, camadas da pilha);

- Implementação de um modelo de energia que permite extrair consumos em diferentes momentos de operação: operação normal e operação perante determinado ataque;
- Modelo de geração de topologias, podendo estas ser definidas como: distribuição aleatória, distribuição em grelha, distribuição controlada (estruturada);
- Mecanismo de introdução de falhas/ataques na rede. Com este mecanismo pretende-se permitir o estudo do comportamento dos protocolos face à possibilidade de introduzir ataques tipificados;
- Utilitários de recolha de dados da simulação, em tempo real e em tempo diferido, que permitam a extracção de medições referentes a propriedades importantes, como consumos de energia, latência, fiabilidade, correcção do protocolo e correcção dos eventos, disponibilizando-os de forma gráfica.

O ambiente de simulação em causa permite estudar, sistemática e comparativamente, diferentes protocolos de encaminhamento seguro em RSSF. Contribui, ainda, com o estabelecimento de uma base de comparação, pelo desenvolvimento de um protocolo, para futuras avaliações de outros protocolos emergentes. Esta avaliação é referente ao impacte que os mecanismos de segurança têm sobre propriedades tão importantes como: o consumo de energia, a cobertura da rede, a latência das comunicações, correcção dos dados e do protocolo. Estas propriedades são, normalmente,extrapoladas da implementação experimental em pequena escala ou da adopção de modelos matemáticos, os quais, devido a variáveis externas, próprias do ambiente de operação, se podem afastar bastante dos resultados reais.

2 . Trabalho relacionado

Este capítulo apresenta uma visão do estado da arte relacionado com a segurança e modelos de simulação em RSSF. A primeira secção apresenta a caracterização de uma RSSF, permitindo realçar as limitações destas redes nomeadamente no que diz respeito ao modelo de energia. A segunda secção apresenta a definição do modelo de adversário e tipologias de ataques. A terceira secção apresenta protocolos de encaminhamento seguro. A ultima secção apresenta diversos ambientes de simulação relacionados com as RSSF e *ad-hoc*.

2.1 Caracterização de uma RSSF

Nas RSSF podem-se considerar vários tipos de nós de computação, estando estes interligados por uma infraestrutura de comunicação textitmulti-hop, em que cada nó da rede (*mote*) pode desempenhar pelo menos três papeis[**?**] : 1) Nó gerador de dados, pela captação da eventos associados às especificidades dos sensores possuídos; 2) Nó encaminhador, que recebendo dados de uns nós os encaminha para outros por forma a que alcancem o destino; 3) Nó de sincronização ou nó de agregação, embora estas duas caracterizações não correspondam à mesma tarefa, por si só, a um nível mais macro, corresponde igualmente a colecção de dados da rede e a um pré-processamento, por forma a fazê-los seguir de forma agregada para outro destino (interno ou externo à rede).

2.1.1 Modelo da plataforma genérica de uma RSSF - *Mote*

Interessa perceber qual a arquitectura inerente a estas plataformas de rede, onde é possível executar uma multiplicidade de aplicações, apesar das limitações impostas pela arquitectura. Na figura seguinte apresenta-se um modelo simplificado[42] que ilustra os diversos componentes que concorrem para o funcionamento de um *mote* numa RSSF.

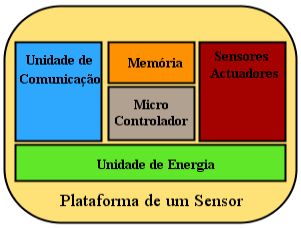

Figura 2.1 Modelo de um Sensor de uma RSSF (baseado em [42])

Como se pode observar na Figura2.1, os sistemas que estão presentes são os seguintes: i) Sistema de processamento; ii) Sistema de comunicação; iii) Sistema sensorial; iv) Sistema de memória; v) Sistema de energia;.

2.1.1.0.1 Sistemas de processamento O sistema de processamento é um componente essencial num *mote*[**?**]. Toda o conjunto de processamento pode ser realizado nas várias arquitecturas representando um *trade-off* entre desempenho, flexibilidade, performance, custo e consumo de energia. Dois dos processadores mais utilizados nas redes de sensores actuais são os da tecnologia Atmel Atmega 128L [**?**] e da Texas Instruments o controlador TI MSP430 [**?**]

2.1.1.0.2 Sistemas de comunicação Este sistema é o responsável pela transferência de dados entre diversos nós de uma rede por intermédio de radio frequência, o que quer dizer sem fios. A rádio-frequência permite ter bom alcance de comunicação com boas taxas de transferência, bem como equilibrar o consumo de energia perante existência de erros.

Os transmissores e receptores [1]nos nós têm a função de transformar uma cadeia de bits, vindas do processador, em ondas electromagnéticas. Os estados pelos quais passam estes componentes são os seguintes e estão associados essencialmente ao consumo de energia: a trasmitir[2], a receber[3], *idle*[4], *sleep*[5]. Alguns dos sistemas de rádio existentes nas redes de sensores são os seguintes: Xemics XE1205 [**?**], Chipcon 2420 (com chipset para 802.15.4)[**?**].

Porque o custo energético da comunicação tem uma ordem de grandeza bastante superior ao do processamento, o existe uma ligação estreita entre o sistema de comunicação e o protocolo

[1]No caso da implementação em RSSF, as plataformas incorporam num mesmo componente as duas funções, transmissão e recepção, estes são denominados por *transceivers*

[2]o transmissor está activo e a emitir dados

[3]O receptor está activo e a receber dados

[4]Está livre para receber dados, note-se que alguns componentes de comunicação estão activos e outros estão desligados

[5]A maior parte dos componentes de comunicação estão desligados

de acesso ao meio (MAC)[27]. Estes protocolos, permitem optimizar o tempo de operação do sistema de comunicação, com vista à redução do consumo de energia permitindo aumentar o tempo de vida útil de uma RSSF.

2.1.1.0.3 Sistemas Operativos Os sistemas operativos (SO's) para as RSSF são menos complexos do que os restantes SO's comuns para PC's. De seguida apresentam-se alguns sistemas operativos com vista a contribuir para a compreensão do estado da arte nesta componente das RSSF.

TinyOS[15] Sistema operativo livre e de código aberto, desenvolvido pela Universidade da California, Berkeley e implementado em nesC (uma extensão da linguagem C) muito optimizado para as limitações de memória existentes nas RSSF. O modelo de execução orientado por eventos possibilita uma maior precisão na gestão de energia, ainda assim tem grande flexibilidade no escalonamento dos eventos gerados pelo ambiente real, que, como se sabe, são de natureza muito imprevista.

Contiki[2] Sistema operativo livre e de código aberto, implementado em C, e tal como o TinyOS[15] é orientado por eventos. As aplicações podem ser carregadas e descarregadas em tempo de execução. Os processos deste SO usam *threads* leves, denominadas por protothreads que proporciona um estilo de programação *threadlike* em cima do *kernel* orientado a eventos.

Nano-RK[10] Sistema operativo desenvolvido na universidade de Carnegie Mellon, o seu *kernel* é baseado em execução em tempo real com *multithreading* preemptivo. Assim é possivel ao *kernel* controlar que processos têm acesso ao CPU, com a divisão em fracções de tempo a que cada processo acede ao CPU, à rede e aos sensores.

2.1.1.0.4 Sistemas de energia A energia é um recurso escasso e como tal deve ser tido em conta no desenho de uma qualquer RSSF, adaptando-se à aplicação ou aos eventos que se pretendem monitorizar. Assim, os dispositivos fornecedores de energia podem ser de diversos tipos, por exemplo: baterias (tipo AA), baterias fotovoltaícas . Encontra-se nas especificações das plataformas mais recentes os valores que indicam o consumo energético nos diversos estados de operação de um *mote* a titulo de exemplo observe-se a plataforma *Mica2*[**?**] da *Crossbows*[**?**]:

Descrição	Valor
Baterias	2xAA
Mínimo V_{in}	2.7 V
Capacidade da bateria	2000 mAh
Regulada	no

Tabela 2.1 Características de energia do *mote* Mica2 (origem:[**?**]))

Operação	Consumo
CPU *sleep* com *timer on*	0.054 mW
CPU activo, *radio off*	36 mW
CPU activo, *radio idle listening*	66 mW
CPU activo, *radio TX/RX*	117 mW
Potência Máxima (CPU activo, radio TX/RX + flash write)	165 mW

Tabela 2.2 Consumo de Energia *Mica2* - Operação Tipica (origem:[**?**])

2.1.2 Modelo de energia de uma RSSF

Nas diversas aplicações das RSSF[**?**], uma das características que tornam estas redes como as mais aconselhaveis para o seu uso é a capacidade de auto-organização, de facilidade de distribuição por locais remotos e de grande dimensão, permitindo a monitorização de eventos durante o maior tempo possível. Assim, para além das preocupações referentes à segurança e correcção de recolha dos dados, a energia é uma problemática relevante e que deve ser considerada em todos os momentos. Desde logo os sensores já possuem, na sua arquitectura, serviços que visam a optimização da energia, permitindo que o tempo de vida útil da rede seja estendido o mais possível. Durante o seu funcionamento, um qualquer nó sensor, passa cerca de 90% do tempo num estado "adormecido", o que implica a inactivação de alguns dos seus componentes com vista a redução do consumo de energia[**?**]. Note-se que este comportamento é possível devido à natureza das próprias aplicações das RSSF, por exemplo, se o número de eventos detectados e propagados pela rede tiver uma frequência de acontecimento muito grande, naturalmente o tempo neste estado de adormecimento terá tendência a ser reduzido. No entanto o comportamento esperado numa RSSF é a ocorrência espaçada de eventos, permitindo por isso que um nó passe a maior parte do tempo num estado em que o consumo de energia seja mínimo.

Existem vários protocolos de acesso ao meio (MAC) que implementam estas mudanças de estado que podem ser completamente autónomas, ou seja não exigindo a coordenação com os

nós vizinhos, ou de forma coordenada em que a informação trocada com os vizinhos contribui para passar mais ou menos tempo em estados de adormecimento[**?**]. Considera-se que existem três estados principais na operação de um sensor: activo, inactivo, adormecido. No primeiro estado, poder-se-ão considerar, ainda, diferentes modos de funcionamento, devido aos componentes de um nó sensor estarem ligados ou desligados em determinado momento e com isso consumirem mais ou menos energia. Na tabela seguinte apresenta-se alguns dos estados considerados numa plataforma do tipo Mica2[**?**].

FALTA TABELA AQUI

A energia nas RSSF é tida em consideração desde a implementação do *hardware* até ao desenho e implementação de qualquer dos serviços existente na pilha de software dos nós sensores. Têm se observado que a maioria dos estudos desenvolvidos são levados a cabo com recurso a ambientes de simulação[**?**, **?**] estes ambientes ou ferramentas que permitem o estudo dos perfis de consumo de energia numa RSSF, são calibrados com recurso às especificações dos fabricantes [] e a resultados de experiências em sensores reais. O estudo desta problemática é ainda mais relevante quando se associa a problemática da segurança na operação das RSSF. Dos componentes de um *mote*, o sistema de comunicação e o CPU são os que mais influenciam o consumo de energia. Ainda o maior peso seja dado para a comunicação, numa ordem de grandeza de 1 para 1000 face ao processamento. Quando se pretendem introduzir mecanismos de segurança ao nível do encaminhamento de dados, é necessário ter presente que existe um custo a pagar no que respeita à energia, devendo-se este facto ao aumento da computação, uma vez que estão envolvidas operações de criptografia[**?**] e mesmo ao nível da comunicação caso os modos de criptografia aumentem o tamanho das mensagens enviadas[**?**]. Conhecidas as limitações impostas pela arquitectura, algumas derivadas da pequena dimensão dos sensores, poder-se-à dizer que os protocolos de acesso ao meio representam um papel importante no consumo de energia[**?**].

2.2 Modelo de Adversário, Ataques ao Encaminhamento e Contra-medidas

2.2.1 Arquitectura de Serviços de Segurança em RSSF

Num sistema seguro é necessário que a segurança esteja integrada em cada um dos seus componentes, não se confinando a um componente isolado do sistema [60]. Nesta secção, apresenta-se, introdutoriamente, alguns requisitos de segurança de uma RSSF e alguns serviços de segurança,

que representam um ponto de partida para a garantia de propriedades de segurança, aquando do desenho de RSSF seguras.

2.2.1.1 Requisitos de segurança de uma RSSF

Os requisitos de segurança de uma RSSF podem variar consoante as especificidades da aplicação que a rede visa suportar. No entanto, apresentam-se, de forma genérica, os principais requisitos de segurança de uma RSSF [60]:

Autenticação Sendo que o meio de comunicação é partilhado, é necessário recorrer à autenticação para garantir a detecção de mensagens alteradas ou injectadas no sistema, de forma não autorizada [60]. O uso de criptografia assimétrica ainda não é viável nas RSSF, considerando as limitações destas redes e as exigências computacionais[6] destes mecanismos;

Confidencialidade Sendo uma RSSF uma infraestrutura baseada, fundamentalmente, na disseminação de dados recolhidos sensorialmente em ambiente remoto e/ou não controlado e, normalmente, de fácil acesso, é necessário garantir a confidencialidade dos dados que circulam na rede. O uso de criptografia é o mais indicado para este tipo de protecção, sendo adequada a selecção de algoritmos de encriptação fiáveis (ex: AES[7] [65], ECC[8][65]. Com a utilização de chaves criptográficas, é necessária a adopção de esquemas seguros de distribuição de chaves [31].

Disponibilidade Entende-se por disponibilidade a garantia do funcionamento de uma rede durante a totalidade do tempo de operação. Os ataques que visam afectar esta propriedade são denominados por ataques de negação de serviço (*Denial of Service - DoS*) [34]. Para além de mecanismos que evitem estes ataques, é necessário garantir que a degradação da rede (na presença de um ataque) é controlada, ou seja, é proporcional ao número de nós comprometidos;

Integridade A integridade garante que os dados recebidos por um nó não são alterados, por um atacante, durante a transmissão. Em alguns casos, esta propriedade é garantida juntamente com a autenticação, usando mecanismos que permitem garantir ambas as propriedades numa só operação. É comum o uso de CMAC [9] [65], uma vez que permite autenticar (com o uso de

[6] Não somente em termos de memória, mas também em termos de energia
[7] *Advanced Encryption System*
[8] *Elliptic Curve Cryptography*
[9] *Cipher based Message Authentication Code*

chave criptográfica simétrica) e verificar a integridade de uma mensagem [61].

Detecção de Retransmissão Ilícita (ou Teste de Frescura da Mensagem) A frescura de uma mensagem garante que não é antiga e/ou não foi reenviada por um atacante [61, 47]. Podem-se considerar dois tipos de frescura: fraca (garantindo ordem parcial e sem informação do desvio de tempo, usada para as medições dos sensores) e forte (que garante ordem total em cada comunicação, permitindo estimar o atraso, sendo usada para a sincronização de tempo).

2.2.1.2 Serviços Básicos de Segurança

Alguns serviços de segurança têm vindo a ser desenvolvidos para as RSSF, com vista a garantir a segurança ao nível da comunicação (ex: criptografia, assinaturas, *digests*). Estes serviços permitem que o arquitecto de sistemas se centre noutras problemáticas relacionadas com o comportamento dos protocolos face a ataques, por exemplo, de intrusão. Apresentam-se de seguida alguns serviços mais comuns que representam as arquitecturas básicas de segurança para RSSF:

TinySec [44] TinySec é uma arquitectura para protecção ao nível de ligação de dados em RSSF. O objectivo principal é o de providenciar um nível adequado de segurança, com o mínimo consumo de recursos. Os serviços de segurança disponibilizados são: autenticação de dados (com a utilização de *Message Authentication Codes*(MAC) [65], em particular o CBC-MAC[10]) e confidencialidade (CBC-MAC). Não implementa nenhum mecanismo que garanta a frescura das mensagens, tornando-o vulnerável a ataques de retransmissão ilícita;

MiniSec [47] Minisec é uma camada de rede concebida para possuir baixo consumo de energia (melhor que o TinySec) e alta segurança. Uma das características principais, que a tornam mais eficiente, é o uso do modo *Offset Codebook* (OCB) [65] para encriptação de blocos. Desta forma, é possível, numa única passagem, autenticar e encriptar os dados, sem aumentar o tamanho da mensagem, contribuindo para um menor consumo de energia. Esta arquitectura tem dois modos de operação: um orientado para comunicação *unicast* (MINISEC-U) e outro para comunicação *broadcast* (MINISEC-B);

[10]Cipher Block Chaining - Message Authentication Code (CBC-MAC))

SPINS [61] Conjunto de protocolos de segurança, constituído por dois componentes principais: SNEP[11] [61] e μTESLA [61, 48]. O primeiro fornece serviços de autenticação e confidencialidade *unicast*, encriptando as mensagens (com o modo CTR[12]) e protegendo-as com um MAC (CBC-MAC). O SNEP gera diferentes chaves de encriptação que derivam de uma chave-mestra, partilhada entre dois nós, com um contador de mensagens para garantir a frescura de cada mensagem. O segundo componente, o μTESLA [61, 48], é um serviço de autenticação de *broadcast*, que evita a utilização de mecanismos mais exigentes, de criptografia assimétrica, recorrendo a criptografia simétrica, autenticando as mensagens com um CMAC;

Norma IEEE802.15.4 [17] Esta norma define a especificação da camada física e de controlo de acesso ao meio das redes pessoais de baixa potência (*LRPAN*[13]). Foca-se, essencialmente, na comunicação entre dispositivos relativamente próximos, sem a necessidade de uma infraestrutura de suporte, explorando o mínimo de consumo de energia. É uma norma que já se encontra implementada em algumas plataformas das RSSF (ex: Micaz [9]). Especifica alguns serviços de segurança [22], representando uma primeira linha de protecção contra ataques à infraestrutura. Estes mecanismos são os seguintes: i) Cada dispositivo mantém uma lista de controlo de acessos (ACL) dos dispositivos confiáveis, filtrando comunicações não autorizadas; ii) Encriptação de dados, partilha de uma chave criptográfica entre os intervenientes na comunicação; iii) Serviço de integridade de cada *frame*, adicionando a cada *frame* um *Message Integrity Code* (MIC) [65]; iv) Garantia de frescura de mensagens (*Sequential Freshness*), utilizando contadores de *frames* e de chaves.

ZigBee [22, 16] Com a norma 802.15.4, orientada para as duas camadas mais baixas da pilha de protocolos das RSSF, a norma ZigBee define as especificações para a camada de rede e de aplicação. Já incorpora alguns serviços de segurança, nomeadamente: i) Frescura, mantendo contadores associados a cada chave de sessão, que são reiniciados em cada mudança de chave; ii) Integridade, com opções de integridade de mensagens que vão desde os 0 aos 128 bits de verificação; iii) Autenticação, ao nível de rede e ao nível de ligação de dados; iv) Confidencialidade, com o algoritmo AES [65] com 128 bits. Esta arquitectura utiliza o conceito de *trusted center* para gestão da segurança na rede, implementando um coordenador de rede ZigBee. Este, acreditado por todos os nós da rede, pode desempenhar três funções: i) Autenticação de nós

[11] Secure Network Encryption Protocol
[12] *Counter Mode*
[13] Low Rate Personal Area Networks

participantes na rede; ii) Manutenção e distribuição de chaves; iii) Segurança ponto-a-ponto entre nós da rede.

2.2.2 Modelo de Adversário

A definição do modelo de adversário permite, desde logo, identificar as características e as capacidades dos atacantes e os ataques que estes podem desencadear na rede. Nesta secção, caracteriza-se o modelo de adversário que enforma este trabalho.

2.2.2.1 Modelo de Dolev-Yao

Um dos modelos de adversário mais conhecidos, quando se trata de análise formal de protocolos seguros, é o modelo de Dolev-Yao [29]. Neste modelo, é considerado que a rede está sobre o domínio do adversário o qual, perante este facto, pode extrair, reordenar, reenviar, alterar e apagar as mensagens que circulam entre quaisquer dois nós legítimos. Com esta assumpção, entende-se portanto, que o adversário transporta a mensagem e, com isso, adopta um ataque do tipo *man-in-the-middle* [65], com comportamento incorrecto. Este funcionamento, entenda-se, não é comparado à intrusão mas sim à intercepção de mensagens e pode ser mitigado pela utilização de mecanismos de criptografia.

As tipologias de ataque consideradas pelo modelo de adversário de Dolev-Yao são instanciadas pela norma X800 [40], que pretende normalizar uma arquitectura de segurança para o modelo OSI [66], através de uma abordagem sistemática para o desenho de sistemas seguros. Esta norma considera a segurança sob três aspectos: ataque, mecanismo e serviço de segurança [65]. O primeiro refere-se à forma usada para comprometer um sistema, por exemplo, alterando ou tendo acesso não autorizado a dados desse sistema[14]. O segundo aspecto considerado são os mecanismos de segurança, que se entendem como o processo que permite detectar, prevenir ou recuperar de um ataque à segurança (ex: encriptação, controlo de acesso, assinatura digital) [65]. Por fim, o terceiro aspecto define os serviços que, fazendo uso de um ou mais mecanismos de segurança, permitem resistir a ataques dirigidos a determinada fonte de informação, quer seja durante o processamento, quer seja durante a comunicação. Considera-se, então, que,

[14] Na literatura, algumas vezes usam-se os termos "ataque"e "ameaça"para denominarem o mesmo efeito. No entanto, recorrendo ao RFC 2828 [63], podemos definir ameaça como uma potencial violação de segurança, ou seja, é apenas uma vulnerabilidade que pode ser explorada para desencadear um ataque. No caso do ataque, trata-se da exploração inteligente de uma ou mais ameaças que resultam na violação, com sucesso, de um sistema que se pretendia seguro

para efeitos da futura dissertação, os ataques subjacentes ao modelo de Dolev-Yao são protegidos a partir do estabelecimento de uma camada básica de segurança, concretizada por uma das arquitecturas anteriormente referidas na Secção 2.2.1.2.

2.2.2.2 Modelo de Intrusão em RSSF

Considerando o estudo de segurança numa RSSF e, dada a sua exposição natural, nomeadamente a física, colocando cada sensor ao alcance de um adversário, torna-se relevante a consideração de novos modelos de adversário. Cada rede pode ser constituída por milhares de sensores e cada um destes sensores é um ponto de possível ataque [59]. Este ataque pode ser tipificado como sendo por intrusão ou captura.

Este tipo de ataques pode ser desencadeado desde o nível MAC [71] até ao nível de intrusão física. Neste último, um actor externo captura um ou mais sensores legítimos e descobre os segredos criptográficos. Este facto permite-lhe replicar [56] os segredos para sensores maliciosos, introduzido-os na rede de modo a que, agindo coordenadamente, possam comprometer a rede. Conseguida a intrusão, o atacante pode induzir, nos sensores legítimos, comportamentos incorrectos, baseados na informação falsa introduzida pelos sensores maliciosos, influenciando o processo de encaminhamento. Estes ataques são de difícil detecção, uma vez que o carácter autónomo das RSSF pode não permitir distinguir um comportamento errado de uma falha. Com a intrusão, um sensor malicioso, embora respeitando o protocolo da rede, pode actuar de forma incorrecta, levando a rede a criar topologias específicas para determinado ataque ou forçando toda a informação a passar por nós maliciosos, podendo estes suprimir ou violar a informação.

2.2.2.2.1 Modelo bizantino: adversários bizantinos O modelo de ataques por intrusão tem algumas parecenças com as denominadas falhas bizantinas [24], que são caracterizadas como falhas arbitrárias com as quais um sistema não está, à partida, preparado para lidar e que se podem traduzir em comportamentos inesperados. Transpondo esta realidade para as RSSF [49], é difícil detectar a introdução de nós maliciosos, autónomos ou replicados a partir de um nó que foi comprometido. No entanto, alguns autores [56, 24] têm-se debruçado sobre esta problemática, a fim de dotarem os algoritmos de encaminhamento com mecanismos que permitam detectar a replicação de nós maliciosos numa RSSF. Para se lidar com ataques com comportamentos bizantinos, recorre-se a mecanismos probabilísticos que, ainda que possam não mitigar o ataque por completo, aumentam a resiliência e acabam por transformar um ataque num mal

menor, definindo até onde pode ser comprometida a rede, por forma a, ainda assim, garantir a fiabilidade necessária para o seu funcionamento.

2.2.3 Ataques ao Encaminhamento

Apesar de existirem ataques que podem ser dirigidos a qualquer uma das camadas da pilha da RSSF, nesta secção apresentam-se os ataques relacionados com a camada de rede, responsável pelo encaminhamento de dados. Os protocolos de encaminhamento em MANETs [25] e em RSSF, de uma forma geral, decompõe-se em três fases: descoberta dos caminhos, selecção dos caminhos e manutenção da comunicação pelos caminhos seleccionados. Os ataques a um algoritmo de encaminhamento, normalmente, podem explorar as vulnerabilidades de cada uma destas fases de forma específica. Em seguida, os ataques são associados à fase do protocolo em que se podem desencadear e são apresentadas, também, as contra-medidas que permitem mitigá-los.

2.2.4 Ataques à organização da rede e descoberta de nós

Nos protocolos do tipo *table-driven* [20], após a descoberta dos nós vizinhos é necessário recolher informação para a construção das tabelas de encaminhamento. No entanto, em protocolos do tipo *on-demand* [20], esta fase é desencadeada em cada início de transmissão. Este funcionamento corresponde à organização e descoberta de nós numa RSSF.

Falsificação de Informação de Encaminhamento Este ataque tem impacte na formação da rede e na descoberta dos nós. Induz a criação de entradas incorrectas nas tabelas de encaminhamento, podendo também fazer com que estas fiquem lotadas e inválidas. Nos protocolos *on-demand*, o impacte pode ser menor, uma vez que obriga o atacante a injectar informação errada a cada ciclo de transmissão. Outro ataque que causa estes mesmos efeitos é realizado por nós atacantes que inundam a rede com pacotes do tipo *Route Request* (RREQ), pondo em causa a disponibilidade da rede.

Ataques *Rushing* O *Rushing attack* [38] é definido pela exploração, por parte do atacante, de uma janela de tempo para responder a um pedido de caminho para um destino. Este ataque é efectivo quando um protocolo (ex: AODV [58]) aceita a primeira resposta que recebe *Route*

Reply(RREP). Explorando isto, o atacante é sempre um candidato a ser o próximo encaminhador, uma vez que não respeita temporizadores nem condições de resposta, podendo depois influenciar o estabelecimento das rotas.

2.2.4.1 Contra-medidas

Os mecanismos de autenticação fazem com que ataques de falsificação de informação ou de inundação de RREQ sejam minimizados. Os nós da rede podem partilhar chaves simétricas (par-a-par) como forma de autenticar as mensagens de dados e controlo do encaminhamento (RREQ e RREP). Desta forma, o atacante, não possuindo a chaves necessárias para a comunicação, não poderá participar no protocolo.

Para fazer face a ataques de *Rushing*, alguns autores [38] apresentam dois mecanismos de defesa: reenvio aleatório de RREQ *Randomized RREQ Forwarding*) e detecção segura (*Secure Detection*). No primeiro caso, cada nó intermédio guarda um conjunto de mensagens RREQ, escolhendo depois, aleatoriamente, um para reenviar. Ainda assim, pode ser seleccionada uma mensagem RREQ maliciosa, daí a existência do segundo mecanismo, que permite a autenticação de mensagens entre dois nós, garantindo que estas pertencem a nós legítimos. Outros mecanismos passam pela selecção de mais do que uma resposta (permitindo que a mensagem seja enviada por outro caminho) ou pela colecção de várias respostas (escolhendo, aleatoriamente, uma para responder).

2.2.5 Ataques ao estabelecimento de rotas

Os ataques desencadeados nesta fase aumentam a probabilidade de um atacante pertencer a uma rota. Estabelecida a rota através de si pode alterar as mensagens ou agir de forma a desencadear ataques na fase de manutenção de rotas.

HELLO Flooding Este ataque explora os protocolos que se anunciam aos vizinhos, emitindo mensagens de *HELLO* [68, 43]. Os protocolos baseados na localização podem ser vulneráveis a este ataque, uma vez que, com um dispositivo do tipo *laptop-class* [43], que possua um alcance rádio suficientemente potente para cobrir toda a rede, é possível anunciar-se a todos os nós como vizinho, forçando a informação a fluir através dele.

Ataque *Sinkhole* No ataque *sinkhole* [54], o atacante, induz os nós da rede a fazerem passar a

informação por dele. Assim, anuncia-se aos nós vizinhos, como tendo boa comunicação com o nó *sink*, tornando-se, assim, um ponto de passagem da informação. O ataque é realizado enviando pacotes de RREQ, alterando a origem e aumentando o número de sequência, como forma de garantir que esta informação se sobrepõe a qualquer informação legítima. Assim, um atacante poderá participar num número elevado de rotas, podendo alterar ou encaminhar, de forma selectiva, a informação. Os protocolos *table-driven* são vulneráveis a estes ataques, enquanto os protocolos baseados em localização não o são, no caso das suas rotas serem estabelecidas *on-demand* [43, 68, 75].

Ataque *Wormhole* Neste tipo de ataque, apresentado por Perrig *et al* [36], dois nós maliciosos colaboram para a realização do ataque. Os atacantes estabelecem uma ligação (em geral, de melhor qualidade) para comunicarem entre si, permitindo a um nó malicioso capturar pacotes ou partes de pacotes e enviá-los pela ligação privada para o outro atacante, noutro extremo da rede. Este ataque é particularmente eficaz em redes RSSF baseadas em localização que, caso sejam comprometidas, não conseguirão estabelecer caminhos maiores do que dois *hops* [74, 68]. Para além disso, os atacantes transformam-se em nós muito solicitados, pois apresentam-se aos outros nós como tendo uma melhor ligação e estando a menor distância do destino.

Ataque *Sybil* Este ataque foi definido como uma acção que permitia atingir os mecanismos de redundância em armazenamento distribuído em sistemas ponto-a-ponto [30]. Outra definição que surge, agora associada às RSSF, é a que o define como "um dispositivo malicioso que ilegitimamente assume múltiplas entidades" [53]. Com estas definições e, devido à sua taxonomia, é um ataque bastante efectivo contra protocolos de encaminhamento [43]. Em particular, nos protocolos que adoptam múltiplos caminhos, o que permite que um nó assuma múltiplas identidades, ocultando o facto de os dados estarem a passar por um único nó malicioso [68, 75].

2.2.5.1 Contra-medidas

Uma das formas de prevenir um ataque HELLO *flooding* [43] é a implementação de mecanismos de resposta (*acknowlege*) a anúncios HELLO. Desta forma, caso o meio de comunicação do atacante cubra toda a rede, um nó legítimo, que não o alcance e, portanto, não receba a resposta, não considerará o anúncio como válido. É possível proceder à autenticação da mensagem, certificando-a numa entidade central que, ao detectar que um nó se anuncia como vizinho de muitos outros nós, toma precauções, repudiando o nó perante a rede [68].

Alguns autores têm vindo a desenvolver algoritmos que visam a detecção de ataques do tipo *Sinkhole* [54]. Um desses mecanismos é o *Sinkhole Intrusion Detection System* (SIDS) [54], orientado para a detecção destes ataques ao protocolo DSR [41]. Este sistema propõe três mecanismos de detecção: i) Descontinuidade de números de sequência (tendo em conta que um atacante tentará usar números de sequência muito grandes, um nó pode identificar os que crescem rapidamente ou os que não respeitam uma ordem crescente, considerando-os um ataque); ii) Verificação de pacotes (os vizinhos podem certificar a origem dos pacotes enviados por um nó. Isto será difícil de realizar nos pacotes atacantes, uma vez que a origem é alterada. Assim, se circularem muitos pacotes não certificados poder-se-á detectar que a rede está sob ataque); iii) Número de caminhos a passar por um nó (cada nó pode observar a sua tabela de encaminhamento e detectar que existem muitos caminhos a passar pelo mesmo nó, logo pode estar na presença de um ataque *Sinkhole* [68, 75]). A utilização de chaves ponto-a-ponto, como forma de garantir que a informação dos pacotes é legítima, evita que um atacante altere dados da mensagem (origem e número de sequência)[70, 28].

A utilização de *packet leashes* [35] permite mitigar o ataque *wormhole*. Assim, existem dois tipos de condições para se aceitar os pacotes vindos de uma origem: a localização e o tempo. A primeira permite que um nó receptor, conhecendo a localização da origem, saiba se um pacote atravessou a rede por um *wormhole*, calculando a distância entre os dois pontos. O segundo baseia-se, essencialmente, no tempo de transmissão do pacote, exigindo, então, a sincronização de relógios. Se for muito rápido a chegar ao destino, este nó assume que está perante um ataque de *wormhole*. A implementação de encaminhamento por múltiplas rotas, também, permite fazer face a ataques *wormhole* [57].

Para o ataque *sybil* em [53, 68], são possíveis dois esquemas de protecção: i) *Radio resource testing* (cada vizinho só pode transmitir num canal, seleccionando um canal para receber e enviar uma mensagem. Os nós que não responderem são tratados como falsos, ao nível MAC); ii) *Random key distribution* (usando um modelo de *key-pool*, são atribuidas n chaves de um conjunto de m. Se dois nós partilharem q chaves então estarão em condições de comunicar de forma segura. Uma função de dispersão, com base no identificador do nó, permite gerar chaves, evitando que um nó possa conhecer uma grande parte das chaves). A noção de reputação dos nós vizinhos pode também permitir detectar comportamentos incorrectos de atacantes *sybil* [70] ou, alternativamente, realizar o anúncio dos vizinhos de forma autenticada [57].

2.2.6 Ataques à manutenção de rotas

Ataque *Blackhole* No ataque *blackhole* [33], o atacante intercepta os pacotes destinados ao nó/área alvo de ataque, informando a origem de que se trata de um caminho de melhor qualidade ou a menor distância forçando todo o tráfego, dirigido ao destino, a circular através dele. Assim, um nó malicioso intermédio, pode anunciar-se com um caminho melhor, apesar de não ter sequer caminho para o destino, originando um "vazio" e interrompendo o processo de comunicação [68, 75].

2.2.6.1 Contra-medidas

Para mitigar os ataques de *blackhole* existem várias propostas [21, 75, 33], das quais se destacam as que implementam os seguintes mecanismos: i) Confirmação do destino, em que é enviada uma mensagem de ACK por cada mensagem recebida, pelo caminho inverso; iii) Definição de limites de tempo de recepção das mensagens de ACK ou de mensagens de falha; iii) Mensagens de falha, geradas quando um nó intermédio detecta o fim do temporizador de ACK; iv) Caminho definido pela origem, significando que, em cada pacote, é indicado, pela origem, o caminho seguido até ao destino. Os mecanismos que não se baseiem em informação qualitativa do caminho também permitem resistir a estes ataques [57].

2.2.7 Ataques por Intrusão/Replicação

Alguns dos ataques tipificados anteriormente podem ser desencadeados a partir de nós maliciosos, [54] introduzidos na rede de forma incorrecta e, posteriormente, replicados. Os ataques por intrusão nas RSSF são tipificados pela capacidade de um atacante se apropriar de material criptográfico que, durante o desenrolar do protocolo de encaminhamento, lhe permita participar nas comunicações, fazendo passar-se por um nó legítimo. Este ataque, quando conseguido, é bastante devastador para a rede, por exemplo, quando utilizado para coordenar um ataque *sybil*.

Os ataques por replicação [56] correspondem à introdução de novos nós na rede, clonados de nós legítimos, mas que possuem comportamentos incorrectos, como seja a transmissão de informação para outros nós atacantes, originando ataques tipificados na Secção 2.2.3. Estes ataques, resultantes da replicação, são particularmente efectivos em sistemas do tipo de votação ou cuja operação da rede dependa de mecanismos de eleição. Pode-se, então, dizer que, mitigar os ataques por intrusão/replicação permite que, à partida, se possam reduzir algumas condições

para a indução de outros ataques.

2.2.7.1 Contra-medidas

Autenticação central Uma primeira forma de defesa contra replicação é a autenticação dos nós, numa estação central, usando os seus identificadores e permitindo detectar inconsistências ou duplicações[28].

Múltiplas Rotas A existência de diversos caminhos para entregar uma mensagem no destino faz com que se possa aliar a tolerância a falhas à resistência a intrusões. Assim, se a mensagem for interceptada por um intrusor, existe alta probabilidade de esta alcançar o destino, usando outro dos caminhos de envio [28].

Detecção de replicação Perrig e Parno em [56] apresentam mecanismos de detecção distribuída de replicações. Um dos mecanismos é de carácter aleatório *Randomized Multicast* e outro, denominado por *Line-Selected Multicast*, serve-se do modelo de comunicação multi-*hop* da rede para detectar nós duplicados. Outro exemplo é o a criação de estruturas auxiliares (ex: árvores binárias), como acontece no protocolo *Clean-Slate* [57].

2.3 Estudo de Protocolos de Encaminhamento Seguro para RSSF

Como ponto introdutório da discussão e apresentação de algoritmos de encaminhamento em RSSF, importa identificar algumas tipologias ou classes destes algoritmos.

2.3.1 Caracterização dos protocolos de encaminhamento em RSSF

Podem-se estabelecer três classes de protocolos [18]: os baseados na localização, os centrados nos dados e os hierárquicos. Os protocolos baseados na localização usam esta informação para tomarem as melhores decisões para alcançarem os destinos (ex: IGF [23]). Os centrados nos dados, ou seja, os que exploram a semântica dos dados, normalmente são baseados em algoritmos que efectuam pesquisas lançadas a partir de nós de sincronização (ex: Directed Diffusion [39]). Por fim, os protocolos hierárquicos, cuja concepção é baseada na construção de grupos de nós, normalmente denominados como *clusters* (ex: LEACH [32]), que funcionam baseados no princípio de agregação de dados do grupo e na transferência da informação para os nós base.

Para além destas classificações, podemos ainda considerar algoritmos quanto ao momento em que são determinadas as rotas de encaminhamento de dados [20]. Assim, consideram-se os protocolos como *table-driven* ou *on-demand*. Os primeiros referem-se a protocolos que mantêm as tabelas de encaminhamento, trocando mensagens de controlo durante a sua operação. Desta forma, observa-se um maior consumo de energia, devido à regular troca de mensagens. No segundo caso, nos protocolos *on-demand*, as rotas são determinadas em cada envio de mensagem. Apesar de acarretar alguma sobrecarga, em cada envio, acaba por compensar em redes mais móveis e com eventos mais espaçados.

2.3.2 Protocolos de encaminhamento seguro em RSSF

Muitos dos protocolos de encaminhamento para RSSF não foram concebidos tendo em conta o factor da segurança [43, 20]. Em vez disso, pretendiam adaptar-se ao ambiente das aplicações e às características e capacidades das RSSF. No entanto, quando se pretende estender a sua utilização para outros domínios, cuja segurança é indispensável, estas preocupações aumentam, uma vez que os mecanismos de segurança implicam directamente um aumento da computação e um aumento no custo da comunicação, reflectindo-se na autonomia dos sensores.

Nesta secção apresentam-se alguns protocolos de encaminhamento seguro em RSSF que visam cobrir todo o espectro da temática deste trabalho.

2.3.2.1 *Secure Implicit Geographic Forwarding* (SIGF)

Uma das formas de abordar o desenvolvimento de protocolos de encaminhamento seguro é implementar mecanismos de segurança em protocolos já existentes, mas que não seguros. Um destes casos é o algoritmo de encaminhamento *Implicit Geographic Forwarding* (IGF) [23], que deu origem a uma implementação segura: o SIGF [70].

O IGF é um protocolo *on-demand*, baseado na localização que, não mantendo o estado ao longo do seu funcionamento, faz com que não seja necessário o conhecimento da topologia da rede ou a presença de outros nós. O seu carácter não determinístico de encaminhamento já representa um mecanismo de segurança perante determinados ataques, mas não é de forma alguma suficiente para manter uma aplicação, com requisitos de segurança, a executar em ambientes críticos.

Funcionamento do protocolo IGF No protocolo IGF o ambiente está definido por coordenadas que permitem a cada nó saber exactamente a sua localização. Com a agregação do nível de rede e do nível MAC[15] num único protocolo *Network/MAC*, é possível [23], no momento do envio do pacote, determinar qual o próximo melhor candidato para encaminhar os dados. O protocolo inicia-se com a origem a enviar uma mensagem do tipo *Open Request To Send* (ORTS) para a vizinhança (com a localização e o destino). Cada nó que se encontre no sextante válido [16] inicia um temporizador de CTS (*Clear To Send*) inversamente proporcional a determinados parâmetros (distância à origem, energia existente e distância perpendicular ao destino), favorecendo os nós com melhores condições. Ao expirar o temporizador, é enviada uma mensagem de CTS que, ao ser recebida, dá início ao envio de mensagens do tipo DATA a partir da origem. Como este protocolo não mantém estado, resiste a mudanças de topologia da rede. O facto de escolher o nó seguinte, em cada envio, constitui um mecanismo de tolerância a falhas que, em caso de ataque, confina os danos à vizinhança do nó comprometido.

[15] *Medium Access Control*

[16] Ângulo de 60º centrado na origem, orientado para o destino e determinado por cada nó, com base na sua localização

Funcionamento do protocolo SIGF [70] A introdução de mecanismos de segurança, num protocolo existente, compreende um acréscimo de sobrecarga no seu funcionamento. Contudo, o protocolo SIGF [70] pretende manter um bom desempenho e uma elevada taxa de sucesso de entrega das mensagens, mesmo durante um ataque. Uma das características deste protocolo é o facto de ser configurável e, como tal, permitir adaptar os mecanismos de segurança ao grau de ameaça. O SIGF apresenta três extensões ao protocolo IGF [23], o que possibilita a evolução gradual de um protocolo seguro, sem estado, para um protocolo seguro, com manutenção de estado, e, com isto, mais pesado e exigente em recursos.

A primeira extensão é a mais simples e a menos exigente em recursos, o SIGF-0. Continua a não manter o estado e a ter um carácter não determinístico. No entanto, não sucumbe a ataques do tipo *rushing* [38], por não emitir logo para o primeiro nó que lhe envie um CTS. Em vez disso, mantém um conjunto de possíveis candidatos a próximo nó. A extensão intermédia, SIGF-1, já mantém estado, mas ao nível local, podendo com isto constituir listas de reputação dos seus vizinhos, por forma a escolher melhor o próximo nó. Por fim, e tratando-se já de um protocolo mais robusto, mas mais exigente, o SIGF-2 partilha o estado com os seus vizinhos. Permite usar mecanismos criptográficos que garantem integridade, autenticidade, confidencialidade e frescura. Acumula as propriedades de segurança das extensões anteriores: SIGF-0 e SIGF-1.

2.3.2.2 *INtrusion-tolerant routing protocol for wireless SEnsor NetworkS* (INSENS)

Este protocolo [28] foi concebido tendo em vista a tolerância a intrusões e, como tal, faz face a uma das tipologias do modelo de adversário preconizado neste trabalho. Para cumprir com este objectivo, foram identificados dois tipos de ataques: ataques por negação de serviço [34] e ataques ao encaminhamento. O protocolo assenta na existência de uma estação base, constituindo-se como um centro confiável, que partilha chaves criptográficas simétricas com cada um dos nós da rede. Esta característica permite que, em caso de comprometimento de um nó, o atacante não terá acesso a mais do que uma chave segura da rede, isolando, de alguma forma, o ataque.

O uso de caminhos redundantes permite aumentar a resiliência a atacantes não detectados, bastando que exista apenas um caminho sem interposição de atacantes, para que as mensagens cheguem ao destino sem serem comprometidas. Note-se que, neste protocolo, não é possível a comunicação directa entre nós da rede, sem que esta não passe pela estação base. O papel fundamental do protocolo, em termos de encaminhamento seguro, é desempenhado pela estação

base. Uma das vantagens, apontadas pelos autores, é a redução das computações nos nós da rede (ex: para geração de chaves, construção de tabelas de encaminhamento), cujas limitações são as conhecidas. A formação das tabelas de encaminhamento divide-se em três fases: Pedido de rotas (*route request*); Recolha dos dados de encaminhamento; Propagação das rotas. A primeira fase corresponde ao envio, por parte da estação base, de uma mensagem destinada a todos os nós da rede, por forma a obter dados sobre as vizinhanças. Numa segunda fase, cada nó envia a sua vizinhança para a estação base. Por fim, depois da estação base tratar toda a informação recolhida, são elaboradas as tabelas de encaminhamento. As tabelas são depois propagadas para cada nó, podendo prosseguir-se com o encaminhamento dos dados, baseado nas tabelas recebidas.

2.3.2.3 *Secure Sensor Network Routing: Clean-Slate approach*

O algoritmo Clean-Slate [57] foi concebido desde o início, de forma sistemática, com características de segurança. É orientado para a comunicação ponto-a-ponto entre nós da rede, visando a resistência mesmo na presença de um ataque (ataque activo). Classifica-se como um protocolo *table-driven*.

Funcionamento do Protocolo Cada sensor da rede recebe um identificador único global, um certificado assinado por uma autoridade de certificação da rede (AR), a chave pública desta entidade e um conjunto de valores (desafios) baseados numa função de dispersão de um sentido (*one way hash function*). Neste protocolo, podem-se identificar as três fases de operação: organização da rede, estabelecimento dos caminhos e manutenção das rotas.

O protocolo estabelece as tabelas de encaminhamento e os endereços dinâmicos (de tamanho variável) para cada nó da rede, usando um algoritmo recursivo de agrupamento, que executa de forma determinística, mediante uma topologia. Os grupos são formados de forma recursiva e hierárquica, até que a rede forme apenas um único grupo. Em cada fusão é acrescentado um bit (0/1) à esquerda, que permitirá distinguir o endereço de cada nó. Dentro de um mesmo grupo, a comunicação é feita usando *broadcast* autenticado, inspirada no protocolo μTESLA [61, 48].

Este algoritmo incorpora mecanismos de detecção de comportamentos incorrectos dos nós, por exemplo, caso pretendam assumir múltiplas identidades (*sybil* [53, 30]). Este mecanismo é desencadeado após a formação dos grupos, com cada nó a anunciar o seu endereço para os vizinhos, aplicando-se um algoritmo de detecção de replicação de nós [56]. Outro mecanismo

para a detecção de formação incorrecta de grupos é a utilização de *Grouping Verification Trees (GVT)*, baseado em tabelas de dispersão que providenciam autenticação ao nível das folhas, usando a raiz para certificação. Cada nó tem uma GVT, permitindo verificar qualquer comunicação trocada com outros nós da rede.

Durante a fase de manutenção das rotas e encaminhamento, o algoritmo incorpora operações que permitem tratar a saída e entrada de nós. Ao detectar a saída de outro, um nó procura, num dos seus vizinhos, um novo nó fronteira, que lhe permita alcançar o grupo antes acessível pelo nó que saiu. A definição de épocas (*ephocs*) permite que, ao fim de algum tempo, o algoritmo de agrupamento se repita, de forma a incluir novos nós. No que respeita ao encaminhamento, o protocolo usa múltiplas rotas, fazendo com que possa contornar áreas comprometidas da rede. Os nós maliciosos são retirados do algoritmo, usando uma técnica denominada por *Honeybee*. Corresponde ao seguinte: quando um nó malicioso (replicado ou não) é detectado, a rede é inundada com um pacote que indica que o atacante deve ser retirado das tabelas e, tratando-se de uma replicação, o nó replicado autosacrifica-se saindo da rede.

De forma sumária, o protocolo Clean-Slate incorpora os três conceitos para o desenho de protocolos de encaminhamento seguro: prevenção (autenticação), resiliência (múltiplas rotas) e detecção/recuperação (GVT/Honeybee). Implementa-os em simultâneo, ao contrário do que acontece com alguns protocolos que apenas implementam um destes conceitos. É, por isso, um protocolo base, indicado para o estudo comparativo com outros protocolos.

2.4 Ambientes de Simulação

Os ambientes de simulação de RSSF surgem como uma necessidade, inevitável, para o teste e desenvolvimento das redes de sensores e de todas as tecnologias associadas [55, 46]. Alguns ambientes têm sido desenvolvidos especificamente para determinados problemas. Outros são adaptados a partir de ambientes já existentes, como é o caso do NS2 [11] ou J-Sim [5], que foram concebidos para simulações relacionadas com redes convencionais (ex: IEEE802.3, IEEE802.11). A característica importante destes ambientes é a capacidade de repetição de experiências, perante as mesmas condições, facilitando, assim, uma análise sistemática do objecto de estudo.

Nesta secção, apresentam-se diversos ambientes de simulação, mais comuns, e que permitam simular um sistema de RSSF. Foram seleccionados, em primeiro lugar, seguindo critérios relacionados com engenharia de *software*. Seguidamente, com vista à avaliação de um ambiente que se mostre adequado para ser utilizado no trabalho de dissertação, foram estabelecidos critérios relacionados com as RSSF.

2.4.1 Critérios Relacionados com Engenharia de Software

Portabilidade da Linguagem Devido às características da linguagem de programação Java, inerente ao seu ambiente de execução, à consequente portabilidade e à programação orientada a objectos, foram seleccionados apenas ambientes desenvolvidos nesta linguagem.

Código Aberto e Livre Esta propriedade permite que se contornem obstáculos inerentes a licenciamento de *software*, ao mesmo tempo que possibilita a análise e aproveitamento de todas as funcionalidades existentes, permitindo introduzir algumas melhorias ou alterações específicas.

Modularidade e extensibilidade Tendo em conta que os ambientes não possuem todos as mesmas características e funcionalidades e, considerando que a componente experimental da dissertação irá introduzir novos mecanismos, o princípio da modularidade e da fácil extensibilidade facilitará o desenrolar do trabalho.

Documentação Alguns ambientes podem não estar bem documentados. Este critério será importante como ponto de partida para o aprofundamento do conhecimento de cada uma das arquitecturas destas ferramentas.

2.4.2 Critérios Relacionados com as RSSF

Escalabilidade da Rede Uma das características mais importantes das RSSF é o conceito de escala, que se deve ao facto de estas compreenderem, normalmente, um grande número de sensores distribuídos por uma vasta área. Assim, é importante que o ambiente de simulação possa suportar experiências com milhares de nós, uma vez que o factor escala é um das propriedades que se querem ver analisadas no trabalho a desenvolver;

Modelo de Colisões/Comunicação Rádio É fundamental que este modelo se encontre presente no sistema de simulação, por ser um componente base das RSSF, relacionado com a camada de acesso ao meio e de ligação de dados (ex: B-MAC, S-MAC) [62, 72];

Modelo de Gestão de Energia A existência de um modelo de energia permitirá adaptar esta funcionalidade e incluí-la na plataforma final, visto tratar-se de uma das propriedades que se deseja estudar.

Capacidade de Emulação Alguns simuladores possuem a capacidade de emular um sensor real, permitindo efectuar o carregamento de código directamente para o *mote*, sem recurso a recompilação. Não sendo um critério mandatório, reveste-se de algum interesse;

Modelo de Mobilidade Ainda que as RSSF sejam maioritariamente instaladas com características estáticas ou de pouca mobilidade, a existência de um modelo de mobilidade poderá possibilitar a avaliação dos comportamentos dos protocolos, mediante esta propriedade;

Interface de Visualização É importante que o ambiente de simulação possua uma interface de visualização, permitindo uma percepção mais fácil dos comportamentos dos protocolos (ex: topologia, cobertura), bem como o controlo da simulação e extracção de resultados.

Modelo de Gestão de Topologia Um factor que pode influenciar o comportamento de um protocolo de encaminhamento é a topologia. Como tal, é relevante a existência deste modelo por forma a avaliar os protocolos desenhados, perante diferentes topologias.

2.4.3 Prowler/JProwler

[6]Esta ferramenta resulta da conversão de um simulador de eventos discretos[17], Prowler [7], implementado em MATLAB, pela Universidade de Vanderbilt, para a linguagem Java. Este simulador pode ser configurado para simular, de forma determinística ou probabilística. Permite a simulação, com diversos nós, podendo atingir os 5000 (ainda que o número possa ser maior,

[17]Fila global, onde são inseridos todos os eventos da rede, tratados sequencialmente ou por prioridade.

por razões de performance, este é o valor máximo aconselhado), usando diversas topologias (dinâmicas/aleatórias), às quais se podem sujeitar diversos algoritmos.

O JProwler modela os aspectos mais importantes do modelo de comunicação de uma RSSF. A natureza não-determinística da propagação rádio é caracterizada por um modelo de rádio probabilístico simples e preciso, que descreve a operação da camada MAC. Possui uma janela de visualização da topologia da rede. Para o desenvolvimento de aplicações ou protocolos são disponibilizadas classes base que se podem estender. Estão presentes dois modelos de rádio: um de Gauss, para topologias estáticas, e outro de Rayleigh, para topologias móveis.

2.4.4 J-Sim

J-Sim (anteriormente conhecido como JavaSim) é um ambiente de simulação baseado em componentes [5], implementado em Java. Não foi desenvolvido inicialmente com vista à sua utilização em RSSF, como é o caso do ambiente SENSE [12], mas o objectivo de extensibilidade é comum. Este ambiente é amplamente usado e implementa um modelo de rede em camadas. No entanto, este simulador não é o mais adequado para o estudo do desempenho em RSSF, visto que este é condicionado pelo *hardware*, pelo sistema operativo, pelos protocolos de rede e pelas aplicações, assim como pelas optimizações específicas entre camadas da pilha de protocolos. Apesar disto, o J-Sim é um importante ambiente de simulação, dada a natureza fracamente ligada dos seus componentes, a qual permite o desenvolvimento e prototipagem de aplicações. Exige, no entanto, algum conhecimento profundo da arquitectura, mesmo para a implementação de protocolos simples.

2.4.5 Freemote

Fremote é uma ferramenta de emulação [4] distribuída[18], desenvolvida em Java, utilizada para o desenvolvimento de *software* para RSSF. O emulador suporta *motes* (Squawk, Sentilla) e plataformas (Java Cards, SunSpot [14]), baseados em Java. Divide a arquitectura em camadas bem definidas por interfaces: Aplicação, Encaminhamento e Ligação de Dados/MAC. Tem um interface gráfico para configuração. Suporta experiências de grande escala (10.000 nós), incluindo a sua integração com nós reais, baseados em Java. Os principais pontos negativos são: i) O modelo de propagação rádio é muito simples, uma vez que não considera obstáculos

[18]Funciona em rede, com a possibilidade de ter diversos clientes de visualização, ligados a um servidor central.

entre os nós; ii) Só existe um modelo de comunicação real, limitado a emulação simples de plataformas específicas (JMote); iii) Não é orientado para a análise de performance das redes, característica que pode ser importante no desenvolvimento de algoritmos para RSSF.

2.4.6 ShoX

A ideia principal deste simulador [13] é a de proporcionar, de uma forma fácil e intuitiva, a implementação e desenho de protocolos de rede, modelos de mobilidade, modelos de propagação de sinal ou de tráfego de rede. Tal como outros simuladores, incorpora um simulador de eventos discretos, que faz a gestão de todos os eventos da rede. Todos os conceitos conhecidos no domínio das redes sem fios são modelados neste simulador (modelo OSI, pacotes, mobilidade e energia). Uma das vantagens é a existência de classes abstractas para reimplementação de novos modelos em cada um dos componentes, facilitando a programação de novos protocolos ou de novas funcionalidades. A comunicação entre componentes é feita por intermédio de eventos, ou seja, não existe acesso de um componente a outro. Deve-se destacar o interface gráfico, que permite operar todas as configurações da ferramenta, sem a necessidade de editar directamente os ficheiros de XML. Para além disso, é ainda possível visualizar a topologia de rede e extrair resultados gráficos da simulação. O facto de o modelo de propagação de sinal ser baseado na norma IEEE802.11 dificulta a adaptação às condições das RSSF. No entanto, a modularidade do sistema permite o desenvolvimento de uma camada IEEE802.15.4 para se aproximar da norma mais recente de comunicação das RSSF[19]. A arquitectura deste simulador aproxima-se bastante daquilo que deve ser um simulador de RSSF, em que as diversas camadas estão bem definidas.

[19]Não cabe no âmbito da dissertação o desenvolvimento do módulo de comunicação

2.5 Discussão e Resumo do Trabalho Relacionado

As RSSF representam um enorme desafio para a investigação de sistemas e protocolos de segurança. As características que as tornam uma mais-valia para a operação em ambientes remotos, apresentam-se, simultaneamente, como as suas maiores vulnerabilidades, em termos de segurança. Este paradoxo é contornado com mecanismos de segurança inovadores e que se distinguem dos existentes nas redes convencionais. Assim, passada em revista as diversas dimensões abarcadas por esta dissertação (protocolos de encaminhamento seguro em RSSF e plataformas de simulação de RSSF), importa apresentar uma visão crítica do trabalho relacionado.

Na Tabela 2.3 apresenta-se uma visão estruturada das contra-medidas que permitem mitigar ou diminuir o impacte dos ataques nas RSSF. O uso de criptografia simétrica é predominante, uma vez que representa uma forma de garantir propriedades, tais como, confidencialidade, autenticidade e integridade. O uso de funções de dispersão de um sentido permite verificar a integridade e, aliado ao uso de "desafios"[20], permite garantir a frescura das mensagens, a um custo computacional reduzido. É de salientar que as implementações destes mecanismos são optimizadas para a redução dos custos computacionais e de comunicação. Esta tabela propicia uma visão mais actualizada das contra-medidas face aos ataques referenciados no modelo de adversário, fruto da consolidação do trabalho relacionado.

Modelos	Ataque	Contramedidas
Dolev-Yao	Ataque ao meio de comunicação	Criptografia simétrica, *One Way Hashing*
Organização e Descoberta da Rede	Falsificação de informação de Routing	Autenticação, *One Way Hashing*
	Ataques de *Rushing*	Selecção aleatória de RREQ, autenticação, verificação bidirectional
Estabelecimento de Rotas	HELLO flooding	Autenticação com verificação bidirectional(*acknowledge*)
	Ataques *Sinkhole*	Autenticação, Distribuição de chaves *pairwise*
	Ataques *Wormhole*	*Packet leaches*, MAC
	Ataques *Sybil*	Distribuição de chaves *pairwise*, selecção aleatória de canais de rádio
Manutenção de Rotas	Ataques de *Backhole*	Definição de temporizadores e mecanismos de confirmação (ACK) autenticados
Modelo de Intrusão	Intrusão	Encaminhamento multi-rota; *One Way Hashing*
	Replicação	Certificação central; Autenticação; Nós vizinhos como testemunhas

Tabela 2.3 Tabela de Ataques *vs* Contramedidas

Tendo em conta as contra-medidas apresentadas, cabe analisar, comparativamente, os algoritmos e a capacidade de resistir aos ataques definidos no modelo de adversário. Assim, na

[20]Números sequenciais, cuja sequência depende da aplicação de uma função $f^n(x) = C$, n vezes, por forma a obter C (*challenge*)

Tabela 2.4, estão marcados com o símbolo ✓ os ataques defendidos por cada protocolo estudado, sendo que os marcados com × não defendem ou só o fazem em condições especiais.

O protocolo SIGF distingue-se dos demais protocolos estudados, particularmente pela sua origem (extensão do IGF) e por ser baseado em localização. Esta característica é específica para determinadas aplicações e obriga à existência de dispositivos especializados nos *motes*. Para além disto, é particularmente indicado para a monitorização de eventos, cuja ocorrência é espaçada no tempo. Por ser configurável, faz com que se adapte consoante o grau de ameaça, aumentando os custos de operação com o aumento da ameaça.

Os protocolos Clean-Slate e INSENS, não necessitam de conhecimento de localização, diminuindo a complexidade da plataforma de rede. Estes protocolos, devido às características que os definem, são excelentes candidatos a um estudo comparativo. Distinguem-se na utilização da uma estação base como unidade central de encaminhamento, no INSENS. Contrariamente, o Clean-Slate tem uma abordagem completamente distribuída. A questão que se levanta, no INSENS, prende-se, essencialmente, com o impacte no consumo energético dos nós próximos (a um *hop*) da estação base, uma vez que esta encaminhará todo o tráfego da rede.

	Ataques ao Encaminhamento							**Intrusão**	**Comunicação**
Protocolos	**Info. Falsa**	***Rushing***	**HELLO flooding**	***Sinkhole***	***Wormhole***	***Sybil***	***Blackhole***	***Intrusão/Replicação***	***Dolev-Yao***
SIGF	✓	✓	✓	×	×	✓	✓	×/×	✓
INSENS	✓	✓	✓	✓	✓	✓	✓	✓/×	✓
Clean-Slate	✓	✓	✓	✓	✓	✓	✓	✓/✓	✓

Tabela 2.4 Tabela de Protocolos de Encaminhamento *vs* Ataques

Ambos os protocolos implementam resiliência à intrusão. Uma das diferenças é que o Clean-Slate tem uma acção preventiva, correspondente à detecção dos nós maliciosos (replicados). Além disso, faz encaminhamento multi-rota, o que minimiza o impacte dos intrusores que resistam à detecção. No caso do INSENS, apenas existe um mecanismo redundante multi-rota.

Em relação aos ataques que podem ser direccionados a cada um dos protocolos, é importante realçar o ataque *sybil*, como um caso de difícil resolução, quando este deriva de uma intrusão, em que um atacante obteve as chaves necessárias para se poder anunciar com qualquer uma das identidades que assumir. Assim, a resistência a este ataque, por parte dos protocolos estudados, não leva em conta este modo de intrusão, porque assume que cada atacante não tem todas as chaves necessárias para se autenticar com falsas identidades e que, como tal, pode ser detectado.

		Ambientes de Simulação				
Critérios		**Prowler/JProwler**	**J-Sim**	**Freemote**	**ShoX**	**Nova Plataforma**
Software	**Portabilidade da linguagem**	Java	Java	Java	Java	Java
	Código Livre Aberto	Sim	Sim	Sim	Sim	Sim
	Modularidade e extensabilidade	Sim	Sim (JTcl)	Sim	Sim	Sim
	Documentação	Apenas comentários no código	Papers e Online	Pouca	Pouca	
Propriedades das RSSF	**Escalabilidade**	Aprox. 5000 nós	Documentado na ordem dos milhares	Documentado na ordem dos milhares	Foram testados 100 nós sem sucesso	Na ordem dos milhares
	Colisões/Comunicação	Sim, modelo B-MAC	Sim, mas 802.11	Sim, mas muito simples	Sim, mas 802.11	Sim
	Gestão de Energia	Não	Não	Não	Sim	Sim
	Emulação	Não	Não	Sim, para plataformas Java *Based*	Não	Não
	Mobilidade	Sim, mas rudimentar	Sim	Sim	Sim	Não
	Visualização	Sim, mas só de visualização da topologia	Existem ferramentas auxiliares	Sim	Sim, mas não em tempo real. Apenas depois de executar a simulação	Sim
	Topologia	Não existe de raiz, pode ser modelado	Não existe de raiz, pode ser modelado	Não existe de raiz, pode ser modelado	Existem alguns de raiz, podendo ser estendido	Existirão de raiz alguns modelos, permitindo a adição de mais

Tabela 2.5 Tabela de Critérios de Avaliação *vs* Ambientes de Simulação

Por fim e sendo a análise de ambientes de simulação um dos focos do trabalho relacionado, apresenta-se na Tabela 2.5 a sua sistematização, contrapondo os critérios aos ambientes estudados. Como uma primeira nota, deve-se salientar que, tendo em vista a concepção de uma plataforma de simulação, a observação dos critérios de *software* para a selecção de um simulador base teve um peso bastante grande, nomeadamente no que se refere à capacidade de extensibilidade reflectida na simplicidade.

Perante a necessidade de seleccionar um ambiente que vise alcançar os objectivos da dissertação, esta selecção recaiu, no simulador JProwler. A sua simplicidade é uma mais valia, uma vez que, sendo composto por nove classes, bem documentadas, é de mais fácil extensibilidade quanto à implementação das funcionalidades requeridas na plataforma concebida. Cada componente de um nó sensor é mapeado numa classe abstracta. O modelo de comunicação é

inspirado no Mica2 com a gestão de eventos baseada no TinyOS [15], o que o torna, do ponto de vista da sua aproximação à realidade, bastante vantajoso. É certo que algumas funcionalidades tiveram de ser implementadas de raiz, como é o caso do módulo energia (existente no ShoX) ou gestão de topologias (existente no ShoX e no Freemote), o que permitirá desenvolver modelos que podem ser bem integrados de forma observar-se melhor as propriedades desejadas. O caso da emulação (existente no Freemote) não é um requisito mandatório da plataforma. Logo, a sua utilização não se apresenta como uma mais-valia, face aos aspectos menos positivos. O Freemote assenta sobre uma interacção cliente/servidor, o que torna difícil a depuração de erros. Possui um modelo de comunicação demasiado básico, não incorporando atenuações do meio ambiente, o que inviabiliza a sua selecção. O J-Sim é realmente uma plataforma poderosa, mas a sua dimensão obrigaria a um esforço adicional, demasiado grande, na medida em que necessitaria de ver desenvolvido muitos dos seus módulos de raiz.

Desta forma, perante a maior complexidade de algumas plataformas, o JProwler constitui uma base simples, eficiente e, simultaneamente, com capacidade para ser enriquecida, de forma a integrar a plataforma de simulação. Ainda assim, alguns dos conceitos presentes nos outros simuladores prestaram significativos contributos conceptuais para o resultado final da plataforma concebida no âmbito desta dissertação.

3. Plataforma para avaliação de protocolos de encaminhamento de dados em RSSF

Tendo sido abordadas as temáticas relacionadas com a problemática da segurança numa RSSF e dos aspectos que concorrem para o seu estudo, nomeadamente no que se refere a ambientes de simulação, neste capítulo apresenta-se a arquitectura da plataforma de simulação desenvolvida no âmbito deste trabalho. Esta plataforma tem como principal objectivo a agilização do desenho de topologias de RSSF, integrando unidades modulares que permitam a sistematização do processo de análise de protocolos de encaminhamento, antes e depois de serem sujeitos a ataques. Esta análise tem em vista o estudo do destes protocolos no que se refere a propriedades de fiabilidade, latência, cobertura da rede e consumo de energia.

3.1 Consolidação da avaliação do simulador JProwler e a sua integração

Na concepção da plataforma de simulação, desde logo foi assumida a necessidade de não ser uma solução *from the scratch*, a decisão passou por estabelecer como ponto de partida um sistema de simulação já existente. Assim sendo, e tendo sido apresentados os ambientes tidos em conta para este estudo, as hipóteses iam desde a implementação de módulos de avaliação de segurança em simuladores mais evoluídos, estando condicionado a um estudo aprofundado da arquitectura de cada um, até à alteração profunda e adaptação, dos simuladores, a ambientes de RSSF, por forma a simular o comportamento dos sensores durante a sua operação.

Perante este cenário e tendo em conta os objectivos da plataforma que se pretende desenvolver, principalmente com o objectivo de integrar componentes orientados para uma avaliação de propriedades de segurança e da forma como estas afectam os protocolos de encaminhamento a decisão foi a utilização do simulador JProwler[6]. Esta escolha deveu-se fundamentalmente ao facto deste simulador ter um comportamento muito semelhante ao de uma rede de sensores uma vez que foi implementado na tentativa de se aproximar o mais possível do sistema TinyOS [15] em particular simulando as plataformas Mica2 [**?**].

O facto do JProwler apresentar uma estrutura minimalista e simples, facilitou o entendimento do seu funcionamento e com isto a identificação de cada componente envolvido na simulação, com vista à sua reutilização/extensão para a incorporação das diversas funcionalidades que se planearam implementar na plataforma. Uma das vantagens da simplicidade do simulador

recaí sobre a necessidade que existe de interceptar determinados eventos que ocorrem durante a simulação por forma a permitir a sua medição. Um exemplo é a necessidade de interceptar o envio e a recepção de mensagens com o intuito de medir os consumos energéticos destas operações.

3.1.1 Arquitectura do simulador JProwler

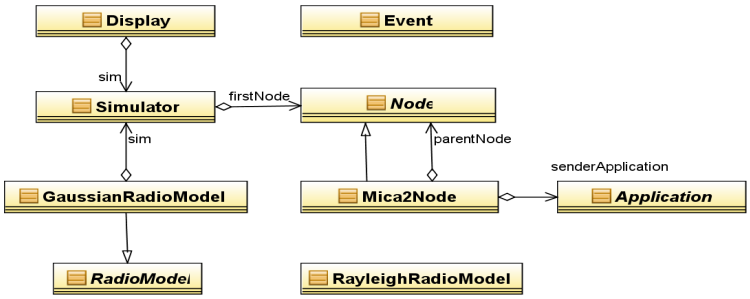

Figura 3.1 Diagrama de classes do simulador JProwler

O simulador base, JProwler[6], é composto por um conjunto de nove classes que são representadas no diagrama 3.1. Este simulador está estruturado em três grupos de funções: a) Motor de simulação, que representa o ambiente de operação de uma RSSF, caracterizado pelo gerador de eventos discretos e modelos de rádio (Simulator, Event, RadioModel); b) Plataformas de execução, que simulam o comportamento dos nós sensores (Node, Mica2Node, Application); c) Visualização, permitindo observar a topologia da rede.

Para utilizar este simulador no estudo de um protocolo de encaminhamento, tal como existe actualmente, é necessário implementar uma classe derivada de Application com a lógica do protocolo. A implementação de uma plataforma sensor, disponibilizada no JProwler[6], é monolítica ou seja não observa uma divisão por camadas, como a apresentada em [**?**]. É certo que nas plataformas físicas observa-se uma forte afinidade entre as camadas do sensor, pois, em alguns casos, ao nível aplicação é necessário ter acesso a dados da camada MAC ou transporte, de modo a facilitar o controlo do estado da aplicação.

Este simulador não incorpora algumas das funcionalidades que se consideram interessantes de observar, de forma genérica, quando se pretende avaliar um protocolo de encaminhamento seguro em RSSF. Um exemplo é a capacidade de ter uma visão do consumo de energia da rede durante uma simulação e poder observar o seu impacte durante a operação normal e durante a

operação sujeita a um qualquer ataque.

3.1.2 Funcionamento do simulador JProwler

Como já foi referido o simulador procura ir ao encontro do funcionamento dos *motes* Mica2[8] e com o sistema de eventos implementado pelo TinyOS[15]. Neste sistema em cada momento apenas está a ser tratada uma mensagem, o que faz com que, caso cheguem várias mensagens ao mesmo tempo, algumas mensagens se possam perder. O modelo de rádio preconiza um mecanismo de *backoff* em caso de falha na transmissão. A introdução de ruído na transmissão é materializada com base numa distribuição de Gauss, este ruído permite aproximar-se do comportamento de uma rede real. A existência de uma pilha de eventos que executa um conjunto de eventos em cada iteração permite simular a ocorrência de diversos eventos em diferentes pontos da rede, ao mesmo tempo.

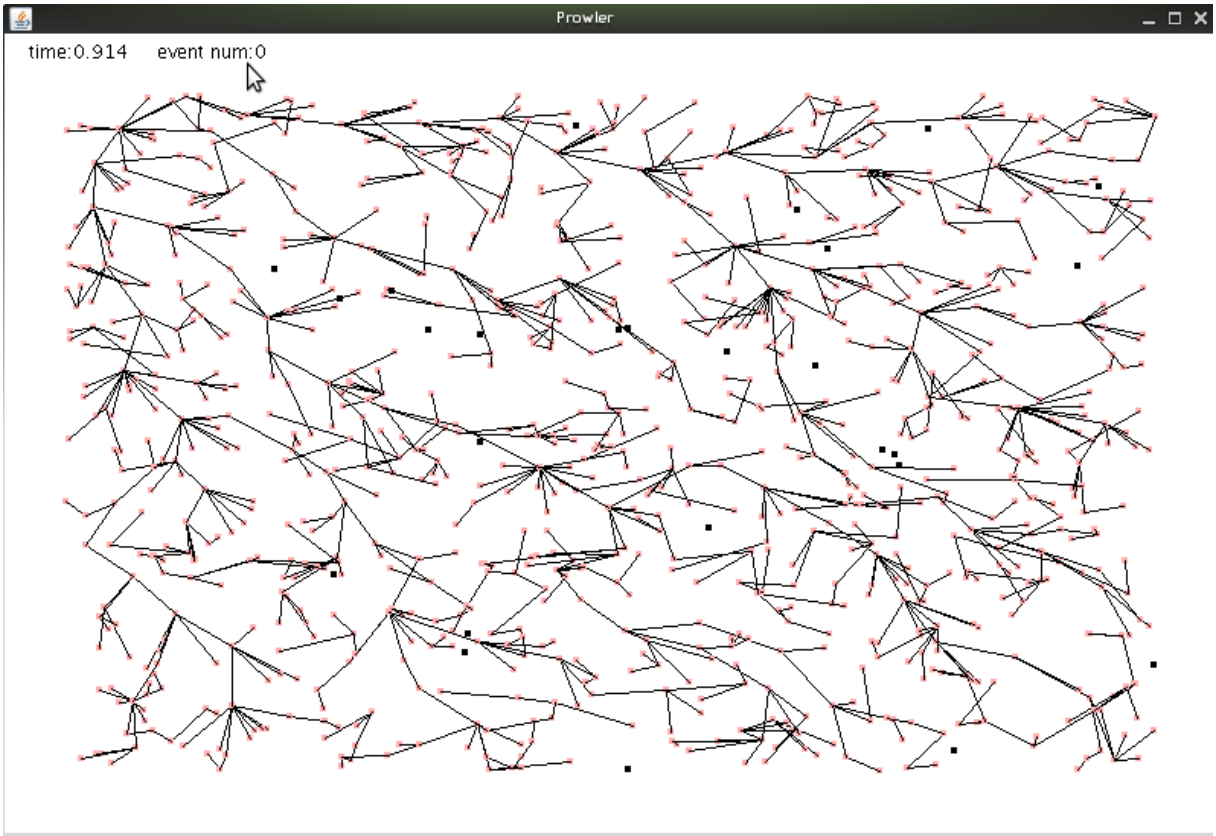

Figura 3.2 Vista do simulador JProwler

A interface gráfica disponibilizada, apenas permite a visualização dos nós sensores e a obtenção de alguma informação referente ao estado da transmissão, materializada por um código de cores, a possibilidade de dispor um determinado nó sensor de forma orientada com recurso a um sistema *drag-and-drop* não se encontra disponível. A distribuição dos nós na área de trabalho é realizado de forma aleatória, permitindo obter uma rede com alguma homogeneidade no que respeita à distribuição dos sensores pela área. A implementação dos sensores, não preconiza a existência de diversos estados de operação dos sensores, ou seja, à semelhança do que se observa ao nível da gestão de energia de um sensor durante o seu ciclo de vida, com os estados, por exemplo, activo, inactivo e adormecido. Estes estados são, normalmente, geridos ao nível MAC dependendo dos protocolos que cada sensor implementa[72, 62].

3.2 Arquitectura da plataforma de simulação

Para melhor percepção da arquitectura da plataforma de simulação, esta é apresentada sob a forma de uma pilha de serviços. Como se pode ver na Figura 3.3, os principais serviços são: i) Simulador base ou motor de simulação; ii) Camada de Instrumentação e Teste; iii) API para a implementação e avaliação de protocolos de encaminhamento; iv) Consola de visualização e controlo de simulação.

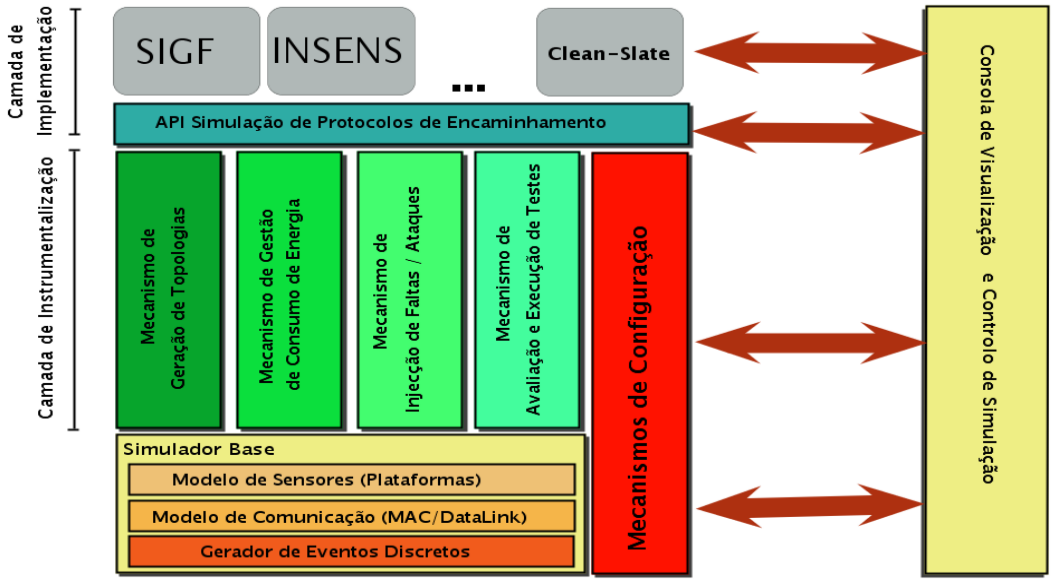

Figura 3.3 Arquitectura de Simulação

3.2.1 Simulador base ou motor de simulação

Como já foi referido o motor de simulação resulta da extensão do simulador JProwler. De uma forma geral, o simulador já implementa os modelos base de um RSSF, nomeadamente no que respeita ao modelo de acesso ao meio e no modelo rádio.

3.2.2 Mecanismo de Configuração

Para dotar a plataforma de maior flexibilidade, a existência de um componente gestor de configurações revela-se importante. Este componente é transversal a toda a plataforma uma vez que os componentes que são passíveis de configuração contêm lógica associada para que as parametrizações possam ser persistentes e portáveis. Foi adoptada a tecnologia XML para a definição dos ficheiros de configuração das simulações. Os ficheiros de configuração da plataforma são implementados usando ficheiros de *Properties* do Java. As principais funcionalidades vão desde as configurações dos parâmetros do simulador base, até à configuração de cada uma das simulações e dos seus componentes, como forma de possibilitar a repetição de experiências nas mesmas condições.

3.2.3 Camada de instrumentação

Esta camada contém um conjunto de ferramentas que controlam a simulação nomeadamente no que respeita aos mecanismos principais que se pretendeu implementar na plataforma:

3.2.3.1 Mecanismo de Geração de Topologias

As RSSF, normalmente, são caracterizadas por diferentes formas de distribuição dos nós sensores. Estas distribuições podem ser essencialmente divididas em dois modelos: aleatório e estruturado. Sabe-se que, de facto, a topologia da rede pode influenciar o comportamento de um protocolo de encaminhamento. No sentido de permitir a análise do impacte de diferentes topologias nos protocolo, a plataforma dispõe, de um componente cuja função é gerar/desenhar topologias de rede.

A plataforma oferece, de forma automática, a geração de três tipos de topologias: 1) aleatória, delimitada pela área seleccionada; 2) em grelha, mediante a introdução da distância entre

cada sensor, também numa área delimitada; 3) orientada, posicionado onde se pretende introduzir um sensor. Os primeiros dois modos permitem estabelecer uma primeira distribuição, que seguidamente pode ser ajustada manualmente, de modo a desenhar uma topologia estruturada específica, através da introdução orientada de sensores, como é o exemplo da estação base. A remoção de sensores também é disponibilizada como forma de customização de topologias podendo se criar, por exemplo, zonas sem sensores para testar determinada topologia.

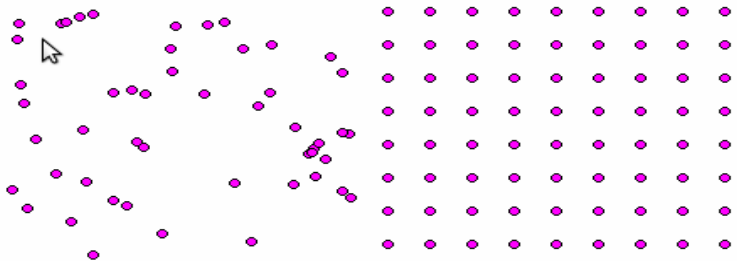

Figura 3.4 Exemplo de dois modos de geração de topologias

Este componente possibilita a sua extensão para adição de novas topologias, geradas de forma automática (específicas para determinada simulação/avaliação). Uma vez que se pretende possibilitar o estudo comparativo de protocolos é dotado de uma mecanismo persistência que possibilita a reutilização, de determinada topologias, em diferentes experiências.

3.2.3.2 Mecanismo de Gestão de Consumo de Energia

Este é um dos componentes de maior importância, uma vez que um dos indicadores que se pretende observar na análise de protocolos de encaminhamento é o impacte sobre o tempo útil de operação da rede, quer em condições de funcionamento normais, quer em condições de ataque efectivo, tempo este que está dependente da energia. A energia é um recurso escasso numa RSSF. Para a implementação de mecanismos de segurança, é necessário recorrer a maior computação, transmissão de dados de forma segura e capacidade de verificação de determinadas propriedades de segurança.

Sendo reconhecida a importância desta ferramenta os princípios orientadores para a sua implementação foram: i) Tornar a expressividade destes consumos na lógica dos protocolos o mais transparente possível; ii) Adoptar um modelo o mais próximo possível da realidade, principalmente no que refere aos consumos de cada operação de um nó sensor; iii) Parametrização do modelo de energia por forma a permitir a calibração o mais próxima do valores reais; iv)

Capacidade para analisar em pós processamento os dados resultantes da análise de consumo energético, para a elaboração de gráficos representativos dos resultados obtidos; v) Capacidade de, em tempo real, prever os perfis de consumo.

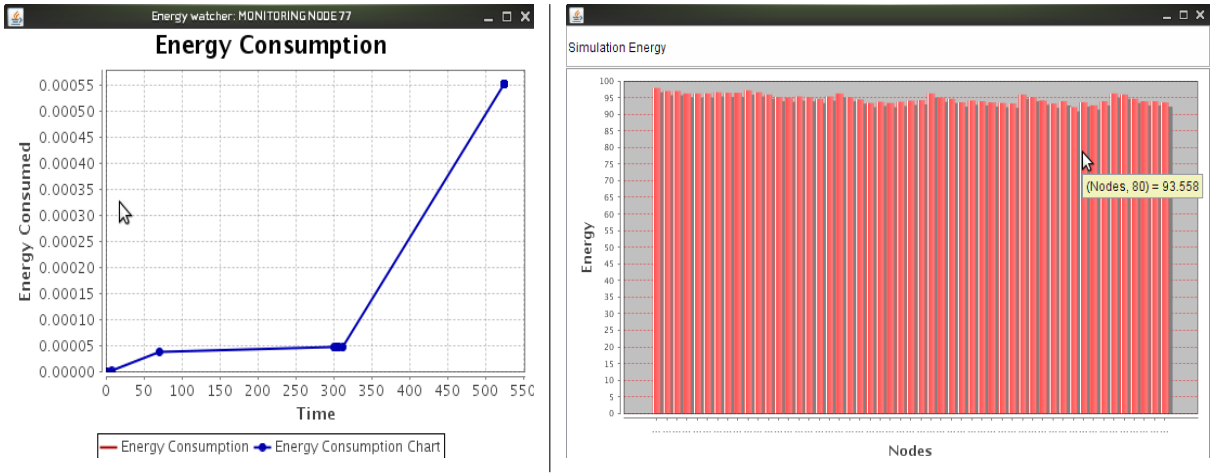

Figura 3.5 Modos de visualização em *runtime* do consumo de energia

Na Figura 3.5 é possível observar, em primeiro lugar, o consumo de um nó específico, em função do tempo. Em segundo lugar, observa-se o estado global da bateria de um conjunto de sensores. Esta visualização contribuí para a afinação da análise e dos testes, nomeadamente nas experiências que possuam execuções mais longas, no fundo permite a ante-visão dos comportamentos dos protocolos, no que respeita a perfis de consumo de energia.

3.2.3.3 Mecanismo de Injecção de Falhas /Ataques

Para que se possa avaliar a segurança nos protocolos de encaminhamento é necessário sujeita-los a ataques que possam expor as suas capacidades e vulnerabilidades de mitigar os efeitos dos atacantes. A não existência de um simulador que permita a indução de ataques de forma generalista, evidencia a pertinência deste componente como factor diferenciador desta plataforma face a outras existentes. Das características que se podem encontrar neste componente evidenciam-se as seguintes: i)Componente flexível permitindo adaptar-se à lógica de cada algoritmo; ii) permite a visualização do código malicioso em tempo de execução, por forma a perceber comportamentos dos protocolos face ao ataque; iii) Permite acrescentar mais modelos de ataques, dos já tipificados neste relatório ou de outros que venham a ser identificados.

Com o mecanismo de injecção de ataques procura-se tornar o mais intuitiva possível a sua

utilização, baseando-se numa API simples e genérica. A integração com a consola de visualização permite identificar visualmente os sensores maliciosos face aos outros sensores. Assim, também é possível activar o modo de ataque em cada um dos sensores bem como seleccionar o ataque que se pretende observar, podendo coexistir diversos ataques. No entanto, está limitada a operação de apenas um ataque de cada vez em cada sensor. No que se refere a avaliação, é possível medir o impacte do ataque pelo número de mensagens que atravessam os nós atacantes, ou pela observação do impacte nas propriedades observadas no âmbito deste trabalho.

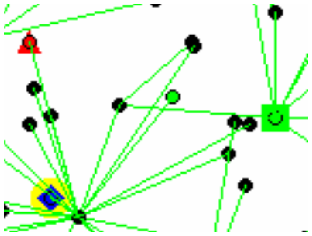

Figura 3.6 Representação de um nó atacante com um triângulo vermelho

3.2.3.4 Mecanismo de Avaliação e Execução de Testes

Uma vez que só se consegue avaliar o que é possível medir, foi necessário proceder-se ao desenvolvimento de ferramentas de medição das propriedades, para além do consumo de energia, que se pretende observar em cada um dos protocolos. A cobertura da rede, a latência das comunicações e a fiabilidade da rede são as propriedades principais para as quais foi necessário implementar componentes específicos para avaliação em cada simulação ou experiência. Com a capacidade de medir instalada, é necessário obter os resultados de cada medição. Esta abstracção é representada por intermédio de um modelo parametrizável de testes e da consequente recolha de resultados associados a cada teste.

3.2.3.4.1 Módulo de Cobertura/Fiabilidade/Latência da Rede Mediante as tipologias de ataques estudadas, observa-se que, uma das resultantes destes ataques é a possibilidade de criação de partições na rede, limitação nas comunicações de modo selectivo ou alteração total ou parcial da informação de encaminhamento, resultando, em qualquer um dos casos, na perda de dados e com isto o comprometimento da finalidade da aplicação. Assim, interessa desde logo observar a capacidade de resistência de um protocolo quando sujeito a determinados ataques e aferir qual o nível de disponibilidade da rede durante estas fases. Não obstante este facto, a

rede deverá sempre ser medida, complementarmente, durante o seu funcionamento normal ou seja não sujeito a ataques. Com este módulo, é possível medir as propriedades de cobertura, fiabilidade e latência da rede seguindo as seguintes as definições:

Cobertura: É entendida como a capacidade de um qualquer sensor da rede conseguir comunicar com um qualquer outro sensor, utilizando o protocolo de encaminhamento implementado. A análise resultante da observação desta propriedade pode ser comparada com a cobertura física estabelecida pela comunicação rádio. Interessa portanto realçar que apesar de todos os sensores de uma rede estarem cobertos em certa medida, por comunicação rádio, isso não é condição suficiente para garantir a transmissão de dados de um qualquer ponto A para um qualquer ponto B, uma vez que podem existir partições na rede, sejam estas causadas por ataques ou por falhas energéticas dos sensores fronteira de cada partição.

Fiabilidade: É entendida como a medida que avalia a qualidade da comunicação face ao modo de transmissão, modo este que compreende toda a camada de software utilizada na arquitectura de um sensor. Assim, o resultado da análise desta medida indica o índice de qualidade de informação que um qualquer sensor pode enviar para outro. É medida tendo em conta o numero de mensagens enviadas, face às mensagens recebidas, entre quaisquer dois sensores de uma rede, tipicamente um sensor gerador de eventos e uma estação base. Alguns parâmetros podem ainda afectar os resultados observados, como por exemplo o número de retransmissões de cada mensagem e o intervalo de tempo entre o envio de cada mensagem, que, quando mal parametrizados podem contribuir para a perda de qualidade de comunicação na rede, por saturação da rede.

Latência: Tratando-se de redes, a latência é uma métrica sempre a ter presente na sua analise. Esta medida pode ser efectuada de duas formas: a) por tempo decorrido no envio de mensagens entre dois pontos; b) por numero de nós percorridos no envio de mensagens entre dois pontos. A primeira pode ser difícil de medir, uma vez que se trata de um sistema simulado, em que o factor tempo nem sempre tem a resolução que permite aferir com rigor a qualidade desta medida. No segundo caso, o número de *hops* percorridos por uma mensagem pode ser uma medida bastante genérica e precisa, que permite aferir esta propriedade da comunicação entre dois quaisquer pontos, podendo-se depois extrapolar um valor estimado para o tempo decorrido em função de uma largura de banda

especificada. No entanto, se se pretender avaliar a possibilidade de utilização de um protocolo em aplicações em que o tempo de resposta é central (aplicações com características de tempo-real, não muito comuns em RSSF), torna-se também importante considerar o tempo como unidade de medida a avaliar.

A avaliação de um protocolo de encaminhamento, preconiza a repetição de baterias de testes e a variação de alguns parâmetros como base de comparação de resultados. Desta forma podem-se observar tendências e comportamentos da rede. O modelo de execução de testes e recolha de resultados segue um modelo que permite sistematizar a sua re-utilização face a diversas distribuições de RSSF e protocolos implementados. Este componente do simulador é importante uma vez que permite agilizar o estudo e concentrar nas experiências o foco da avaliação de um protocolo de encaminhamento de dados. Assim sendo, compreende-se como experiência, uma simulação com as seguintes propriedades:

1. Topologia bem definida (aleatória, em grelha ou estruturada);

2. Protocolo de encaminhamento de dados;

3. Parametrização do modelo de energia;

4. Bateria de testes;

5. Ataques ao nível do encaminhamento de dados;

Estas propriedades permitem sistematizar a avaliação de um qualquer protocolo de encaminhamento de dados seja ele seguro ou não. Para avaliar quaisquer protocolos é possível cruzar estas propriedades de diversas formas e comparar os resultados com uma abordagem metódica. O que se observa em alguns simuladores é que a forma de implementação dos testes é diferente em cada protocolo o que pode levar a diferenças que influenciem os resultados. Com a definição destes critérios e da metodologia genérica de avaliação de protocolos é possível, efectivamente, ter a mesma base de comparação para protocolos diferentes, permitindo afinar as conclusões no que concerne ao seu comportamento em termos de segurança e no impacte nas propriedades que se pretende observar.Esta abordagem transforma-se assim num contributo de inovação para o estudo de protocolos de encaminhamento em RSSF, observando também as características de segurança.

3.2.4 Consola de Visualização e Controlo de Simulação

Como componente integrador de toda a plataforma foi desenvolvido uma consola de operação, que permite interagir com o simulador de forma intuitiva e fácil, facilitando o processo de experimentação durante todo o seu ciclo. Como tal, foi implementado um componente de visualização gráfica de toda a simulação, com capacidade de controlo de parâmetros de execução e configuração, obtenção de resultados da execução de testes.

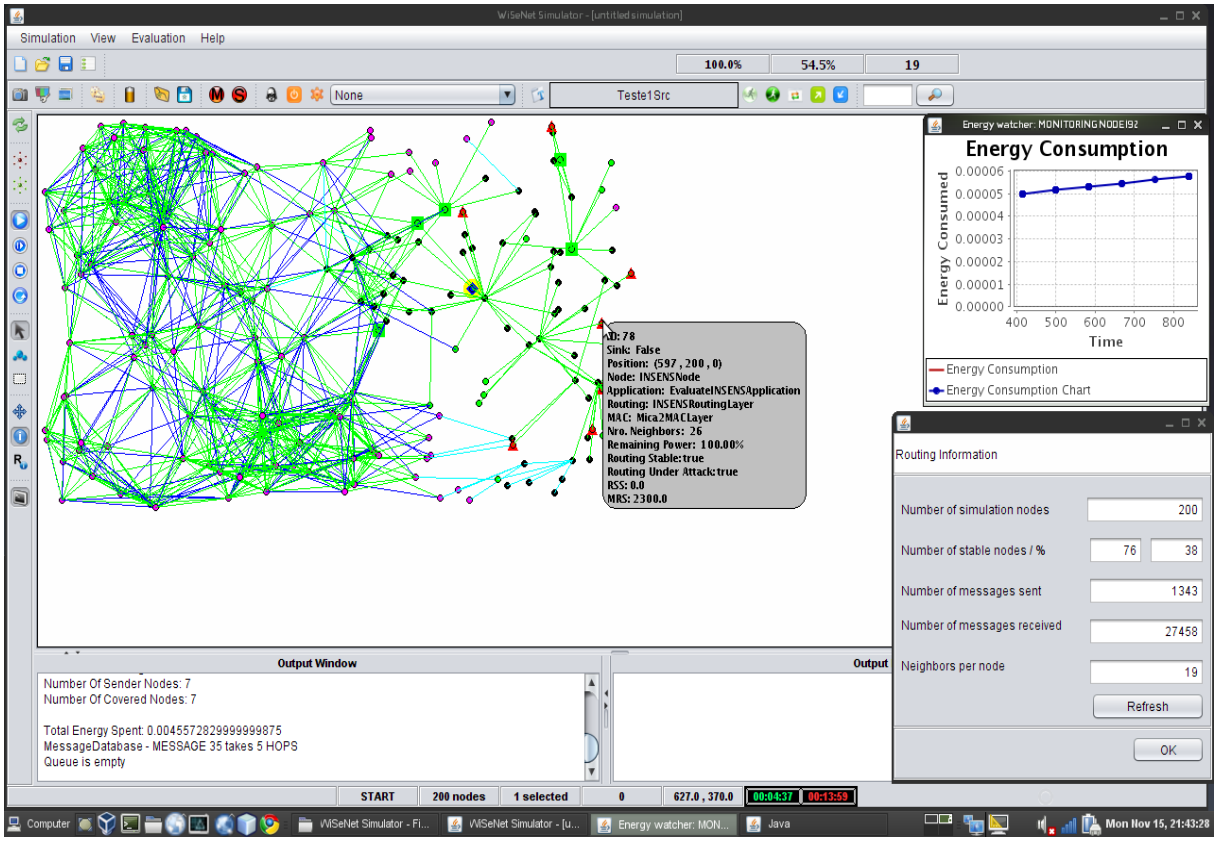

Figura 3.7 Vista do simulador em execução

A Figura 3.9 apresenta uma imagem do simulador durante a execução expondo algumas das funcionalidades disponíveis para a avaliação de protocolos.

3.2.5 API para implementação de protocolos de encaminhamento

Uma vez que o ponto de partida da plataforma é uma simulador que respeita uma API, com a integração de novas funcionalidades com características de avaliação em diferentes camadas do *software* das plataformas sensoriais, surge um conjunto de interfaces que garantem a inter-operação entre os protocolos e o simulador facilitando a medição das propriedades já indicadas. No capítulo seguinte é possível observar alguns destes interfaces que tornam a utilização de algumas funcionalidades o mais transparente possível por forma a permitir o mesmo tipo de acções de avaliação, que contribuem para uma melhor sistematização da análise entre diversas implementações de protocolos.

3.3 Metodologia de utilização da plataforma

Pretende-se com esta plataforma contribuir para uma metodologia de avaliação de protocolos de encaminhamento, principalmente, facilitando a comparação de diferentes protocolos. Assim, assumiu-se desde cedo a necessidade de estabelecer um modelo de utilização da plataforma, modelo este que não é obrigatório para a realização de avaliações com esta ferramenta. Sendo, no entanto, o utilizado na prova de conceito, tendo-se constatado que facilita uma abordagem iterativa de análise. com vista a melhorar/refinar os resultados obtidos. Esta metodologia tem três fases, que serão descritas sumariamente de seguida, sendo que a segunda e terceira fase podem ser feitas uma vez e usadas em diferentes protocolos estabelecendo-se logo como uma base de comparação para outros protocolos que se pretenda estudar.

3.3.1 Implementação de protocolo

Esta fase compreende a implementação da especificação do protocolo em estudo. A utilização das facilidades disponibilizadas pela API e o cumprimento dos interfaces requeridos para a instrumentação do simulador representam as únicas obrigatoriedades na implementação. Uma vez feito isto, algumas funcionalidades apresentam-se disponíveis sem que se tenha feito qualquer esforço adicional, como é o caso, por exemplo, dos consumos energéticos de transmissão, recepção e mudança de estados dos sensor. No Capítulo 4 serão apresentados mais detalhes quanto a esta fase.

3.3.2 Caracterização do ambiente de simulação

A caracterização do ambiente de simulação, refere-se essencialmente ao ambiente de operação que se pretende estudar. A definição de características de operação como por exemplo: 1) alcance rádio; 2) topologias de rede; 2) atenuação radio ambiental; 4) altitude do terreno; 5) definição de nós base e a sua localização; permite estabelecer um modelo de simulação que persiste entre diferentes experiências, retirando algum custo de configuração e permitindo uma abordagem incremental pela realização de pequenos ajustes resultante da observação de resultados face a comportamentos não esperados.

3.3.3 Execução/análise de testes e resultados

A realização de testes, através do envio de mensagens em nós distintos da rede, foi a abordagem metodológica estabelecida para avaliação de um protocolo de encaminhamento em RSSF. Seguindo o princípio de comparação entre diversos protocolos a persistência destes testes permitem a criação de baterias específicas de testes quem concorrem para o melhoramento das análise e a construção de uma base de comparação para futuras implementações. Os parâmetros existentes num teste são os seguintes:

- Número/% de sensores emissores;
- Número/% de sensores receptores;
- Número/% de sensores atacantes;
- Número de mensagens enviadas;
- Intervalo de tempo entre cada envio;
- Número de retransmissões;
- Ataque activo;

A figura seguinte apresenta a janela que possibilita a especificação de testes tendo por base os parâmetros enunciados podendo-se observar a facilidade existente para tornar estes testes persistentes. Uma vez executado o teste importa observar os resultados por forma a coligi-los para análise gráfica dos dados.

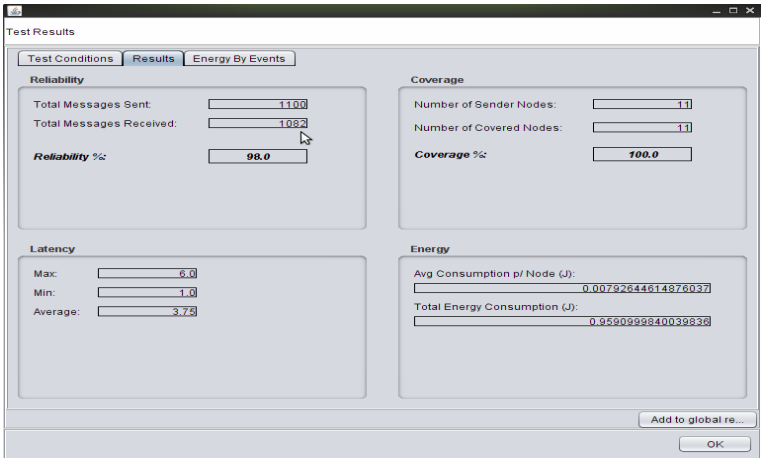

Figura 3.8 Janela resultados da execução de um testes

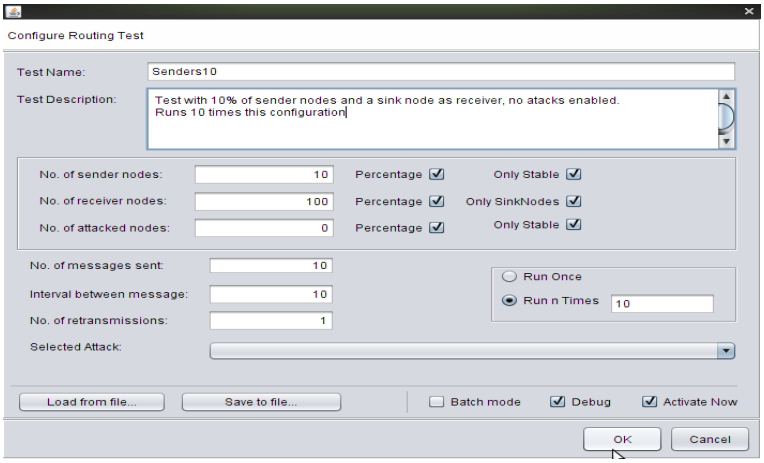

Figura 3.9 Janela para a construção/selecção de testes

Com este modelo de testes entende-se ser possível sujeitar os protocolos a condições que permitem levar a cabo a medição em cada topologia e ataque por forma a facilitar o seu estudo e a sistematizar as experiências. Uma vez que a implementação de toda a plataforma foi realizada em Java será sempre possível estender as funcionalidades por forma a incorporar novos modelos de avaliação e novas abordagens ou metodologias que enriqueçam o estudo das RSSF, nomeadamente no que concerne à problemática da segurança de protocolos de encaminhamento de dados, funcionalidade representada pela figura X.

4 . Implementação da plataforma de simulação

Neste capitulo é apresentada a implementação do simulador. Inicialmente apresentam-se as principais funcionalidades, de seguida descrevem-se alguns detalhes dos componentes mais importantes, quer pela apresentação de diagramas, quer pela apresentação de alguns excertos de código. O projecto está disponível em código aberto, conta com cerca de 236 classes e 31000 linhas de código em Java (JDK 1.6). É possível aceder de forma gratuita a partir do sitio: http://code.google.com/p/secwsnsim/.

4.1 Principais funcionalidades

As principais funcionalidades que se encontram na versão actual da plataforma que se encontra disponível no sitio acima referido são:

4.1.1 Portabilidade

Uma vez que foi integralmente desenvolvido em Java, pode ser executado em qualquer plataforma independente do sistema operativo. O ambiente gráfico foi desenvolvido com recurso a um tema nativo por forma a manter o mesmo aspecto em qualquer sistema operativo.

4.1.2 Extensibilidade

Alguns padrões da engenharia de software foram utilizados, com vista a facilitar a extensão de funcionalidades do ambiente de simulação. Foram usados padrões de *Factories*(para criação de objectos de forma generalizada), *Singleton* (facilitar o acesso a funcionalidades transversais) e *Publish-Subscribe* (notificação de eventos, por exemplo: Notificação de inicio e fim de execução da simulação).

Os componentes mais importantes da RSSF foram mantidos abstractos por forma a permitir especializações dependendo do protocolo/aplicação que se pretende avaliar (exemplo: *MACLayer, RoutingLayer, RadioModel, Application, Message*, etc).

4.1.3 Usabilidade

Com a existência de um ambiente gráfico rico, é possível controlar todo o ciclo de simulação a partir da consola de visualização. De uma forma geral é possível:

Distribuição de sensores Possibilidade de movimentar sensores, adicionar ou remover um grupo ou apenas um sensor numa área seleccionada;

Estados e informação dos sensores Possibilidade de consultar/visualizar o estado e informação de cada sensor (exemplo: posição, altitude, energia) e alterar estado de um sensor individualmente ou de um grupo de sensores (exemplo: ligar/desligar, alterar altitude, activar visualização de vizinhos);

Pesquisa de sensores Uma vez que uma simulação pode ter milhares de nós é possível encontrar qualquer sensor com base no seu identificador único;

Controlo da simulação Possibilidade de iniciar/parar/terminar uma simulação através da consola de simulação.

Depuração de erros/mensagens Visualização integrada de mensagens/erros enviados usando a primitivas Java para a consola de sistema, para a consola de visualização;

Construção de uma simulação Disponibilização de uma ferramenta para desenho de configurações para simulação;

Construção de testes Disponibilização de uma ferramenta para especificação de condições de testes que permitem avaliar um protocolo em estudo;

4.1.4 Facilidades

1. Capacidade para fazer persistir algumas das configurações entre simulações por forma a permitir a reutilização em diferentes protocolos;

2. Introdução de imagens de fundo para simulação de ambientes de distribuição de redes;

3. Definição de parâmetro relacionado com o ambiente permitindo simular a indução de comportamentos derivados do meio ambiente (ex: atenuação causada pela humidade do ar nas comunicações rádio, altura do solo a que um sensor esta colocado);

4. Funcionalidade para tirar fotografia da rede, permitindo retirar uma imagem do estado da rede tal como ele é visto na consola de visualização;

5. Customização das cores de visualização da consola por forma a ajustar face a diferentes imagens de fundo;

6. Obtenção de resultados dos testes executados;

4.2 Implementação dos componentes da plataforma

Nesta secção serão dados alguns detalhes ao nível da engenharia de software expondo o modo como foram implementados alguns componentes e alguns princípios tidos em conta no seu desenvolvimento. Realça-se fundamentalmente a implementação dos seguintes componentes: i) modelação dos sensores; ii) modelo de energia; iii) modelo de medição; iv) modelo de testes e resultados v) API para a implementação de um protocolo.

4.2.1 Representação da plataforma sensor

A versão JProwler original implementa , como já foi referido uma visão mais monolitica da plataforma sensor sem a separação dos papeis de cada uma das camadas da pilha de software de um *mote*. Assim sendo a melhor forma de estruturar um nó sensor é seguir um modelo de camadas, o que levou à separação de cada um dos componentes que se encontravam consolidados numa única classe representativa de um nó sensor. Esta visão permitiu a separação de conceitos, sendo consideradas apenas três camadas: i) Camada MAC; ii) Camada de Encaminhamento; iii) Camada de Aplicação. Na figura 4.1 é possível observar esta organização.

De forma tornar o nó sensor ainda mais modular procurou-se mapear os componentes de um sensor num conjunto de classes, tal como é apresentada na figura 4.1.

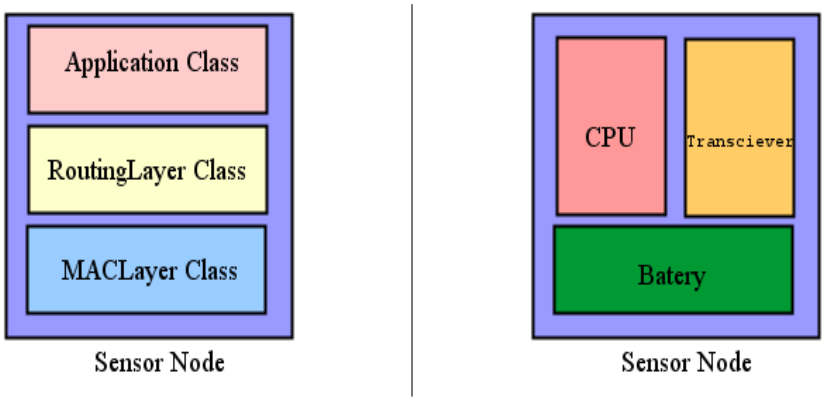

Figura 4.1 Pilha de serviços do sensor e organização por componentes

No que se refere à noção de nó sensor também se pretende que este tenha um desenho bastante semelhante à arquitectura real. Para tal, o nó sensor é modelado por componentes equivalentes à realidade física. Isto é, existe um componente *CPU*, um componente *Transciever* e um componente *Batery*. Este desenho permite que os dois primeiros componentes interajam entre si e em particular com o terceiro para cálculo do consumo energético. Assim, cada componente tem associado um conjunto de acções que são da sua responsabilidade em que cada uma consome um determinado valor energético (definido por parametrização do modelo de energia).

Implementação de uma semântica de *timer* semelhante aos *temporizadores* de tinyOS

4.2.2 Módulo de energia

Com o módulo de energia pretende-se um componente que permita aferir os comportamentos de uma RSSF no que respeita ao consumo de energia. Este componente partiu de um estudo prévio de trabalhos publicados sobre o tema que permitiram coligir alguns valores para afectar em diferentes eventos que ocorrem numa plataforma sensor durante a sua operação.

Primariamente foi elaborada uma tabela que representa os eventos sobre os quais se pretende medir os custos de energia durante a operação de um sensor. Esta tabela resulta da leitura de bibliografia dedicada ao tema da energia nas RSSF, e procura enformar o estudo desta problemática resumindo a informação de forma estruturada, permitindo uma visão geral do consumo de energia num nó sensor. Assim, a tabela de eventos encontra-se organizada da seguinte forma: Na primeira coluna surge a denominação do evento; Na segunda coluna procura-se caracterizar o evento quanto ao custo; Na terceira coluna indica-se a parametrização de referência e por fim na quarta coluna apresenta-se o calculo do custo energético.

Eventos	Caracterização (Custo)	Parametrização de Referência	Cálculo de Custo Energético	Obs.
Transmissão	Varia com a distância (potência aplicada) e com o payload do pacote. CPU=ON TX=ON	µJoule/bit mA - Consumo	µJoule * nº bits Pode ser calculado por tempo de transmissão, com base da largura de banda	Pode ser condicionada pelas condições externas (humidade, altitude) influenciando as retransmissões
Recepção	Consumo semelhante ao da transmissão	µJoule/bit	µJoule * nº bits 0.67µJoule /byte	
Sleep/Listen	Implementação de espera de baixo consumo vs espera activa em modo LISTEN Sleep: CPU=OFF TX=OFF	Tempo*µJoule	T*µJoule 3 picoJoules	Consumo apenas associado a Timers para mudança de estado O estado Listen tem um custo elevado daí a adopção do LPL
Processamento	Numero de ciclos e custo por ciclo	Nº de Ciclos de relógio para processar 1 bit		Difícil avaliação em simulação devido a não aferição do número de ciclos por operação.
Cifrar	Depende a *ciphersuite* (custo computação); Tamanho da chave ; PT=CT ou PT=CT+[OVERHEAD] TX=OFF CPU=ON	µJoule/bit	µJoule * nº bits AES128-> 1.62µJoule *bytes	
Decifrar	Idem	µJoule/bit	µJoule * nº bits AES128-> 2.49µJoule *bytes	
Assinatura/Digest	Comprimento fixo, custo de computação	µJoule/bit	µJoule * nº bits SHA1-5.9µJoule *bytes	Pode optar por redução do numero de bits da assinatura/digest
Verificação	custo de computação	µJoule/bit	µJoule * nº bits	
Transição ON/OFF	Custo de operação dos componentes electrónicos	Fixo, depende do sensor (fabricante)	µJoule * tempo	Tabelas de espicificações do fabricante

Figura 4.2 Tabela resumo com os eventos para medição de custos energéticos

4.2.2.1 Algoritmo do modelo de energia

```
SE NODE==SLEEP ENTÃO
  não detecta eventos (camada aplicação não opera ao nível do simulador)
SENAO
  SE for detectado um EVENTO ENTÃO
      PROCESSA_MENSAGEM()
(1)     SE TX == ON ENTAO
          Transmite
          Consome energia de trasmissão baseado na unidade de referência(J/Byte)
      SENAO (TX==OFF)
          TX OFF==>ON
          Consome Energia de transição OFF==>ON
          Transmite
          Consome energia de trasmissão baseado na unidade de referência(J/Byte)
     FIM SE
  SENÃO (Operação em RELAY ou RECEPÇÃO)
      SE receber uma mensagem ENTAO
          SE TX==ON ENTÃO
             Recebe Mensagem
             Consome energia de Recepção baseado na unidade de referência
                   (J/Byte)
             PROCESSA_MENSAGEM() e TRASMITE Caso se aplique
      SENAO (TRASMISSÃO RELAY)
          (1)
      FIM SE
  FIM SE
FIM SE}

PROCESSA_MENSAGEM(){
  SE estiver a CIFRAR ENTAO
      CIFRA MENSAGEM
      Consome energia de CIFRA baseado na unidade de referência
      (J/Byte) do algoritmo utilizado
  SENAO
      SE estiver a DECIFRAR ENTAO
          DECIFRA MENSAGEM
          Consome energia de DECIFRA baseado na unidade de referência
          (J/Byte) do algoritmo utilizado
      SENAO
          Computacao simples
          Consome energia de COMPUTAÇÃO baseado na unidade de referência
          (J/Ciclo de relógio)
      FIM SE
  FIM SE
```

De entre os eventos apresentados poderemos dividir em dois grandes grupos os que estão associados a estados de processamento e os que estão associados a estados de comunicação. Tendo a segurança como ponto central, não se podem descurar os eventos associados ao processamento,

nomeadamente os que se referem a operações criptográficas, quer estes sejam de assinatura, de cifra ou de integridade. No entanto, a introdução destes mecanismos de segurança contribuem para um aumento das comunicações, uma vez que, normalmente, é necessária informação complementar para verificação de determinadas propriedades de segurança. Vejamos, por exemplo, no protocolo INSENS[28] o facto de todas as mensagens terem um MAC[**?**] o que faz com que o *payload* das mensagens aumente numa razão considerável.

Na caracterização do custo energético dos eventos, existem fundamentalmente duas abordagens típicas[**?**, **?**]: uma que considera a unidade de tempo como unidade de razão para o consumo, por exemplo, calculando o tempo que se passa a transmitir/receber, uma vez que a unidade *joule* está relacionada com o tempo. Outra abordagem é a caracterização com base no tamanho da mensagem calculando com base na energia que se consome por byte operado. No entanto no modelo apresentado a decisão passou por implementar a segunda abordagem uma vez que tratando-se de tempos que não correspondem ao comportamento real num sensor a sua medição tornaria os resultados mais afastados dos resultados esperados. Ao se optar por uma medida baseada em bytes, recorrendo à bibliografia de referência poder-se-à sempre ter os valores adequados aos tamanhos das mensagens.

A forma encontrada para permitir a modelação da energia e a sua implementação de forma o mais transparente possível para um programador que deseje testar um algoritmo na plataforma é a implementação de um interface, que obriga à implementação de uma operação de execução e uma operação que devolve o numero de unidades operadas. Em relação ao número de unidades operadas, esta operação terá a semântica que se pretender tendo por base os valores de consumo parametrizados no modelo de energia. Por exemplo, vejamos o exemplo seguinte para um processamento de cifra de um *payload* de uma mensagem.

```
1.        n.getCPU().executeEncryption(new EnergyConsumptionAction() {
2.          public void execute() {
3.            // execute Skipjack encryption on message payload
4.                byte[] data= SkipjackEncrytion(node.getPayload(), myKey, myIV);
        ...
5.            }
6.            public int getNumberOfUnits() {
7.              return node.getPayload().length; // return de size of payload encrypted
8.            }
9.        });
```

Na linha 7 é devolvido o tamanho do payload em bytes, uma vez que foram esses os dados

que foram cifrados. No entanto o mesmo interface poderia ser usado para devolver nas unidades o numero de ciclos de CPU, se os valores de energia e de ciclos usados para uma cifra Skypjack fossem parametrizados. Desta forma, procurou-se flexibilizar o modelo de energia, primeiro dando liberdade para definição dos valores de consumo com base na unidade de operação desejada, e com a implementação do interface afectar ao componente respectivo a acção de consumo de energia, no exemplo, linha 4 pode-se observar que será do CPU a responsabilidade de consumo de energia. Caso se tratasse de uma transmissão ou recepção seria um consumo a ser executado pelo componente *transciever*.

4.2.2.2 Funcionamento do Módulo de Energia

Depois de identificados os eventos, definidos os valores de consumo de energia para cada um e a formula de cálculo, foi introduzido uma operação no componente Batery para o consumo de cada um destes eventos. Estas operações contabilizam o consumo consultando a parametrização de referência do modelo de energia. Cada evento tem uma etiqueta, que é usada para o registo do consumo e do valor consumido em cada instante em cada um dos nós envolvidos na simulação. Todos os eventos de consumo são registados num repositório para análise gráfica no âmbito da simulação. O funcionamento do módulo de energia resulta da interacção de diversos componentes do simulador, o diagrama que se apresenta a seguir procura representar esta interacção para o exemplo de transmissão de uma mensagem. Observando o código é visível a expressividade para a exploração do modelo de energia e de forma transparente. É necessário parametrizar os valores do modelo de energia e indicar as unidades de consumo.

```
if (isChannelFree(noiseStrength)) {
        \\ start transmitting
    stransmitting = true;
    final Node node = getNode();
    getNode().getTransceiver().executeTransmission(new EnergyConsumptionAction() {

     DefaultMessage m = (DefaultMessage) node.getMessage();
     public void execute() {
       transmitMessage();
     }
     public int getNumberOfUnits() {
            return m.size();
        }
  });
}
```

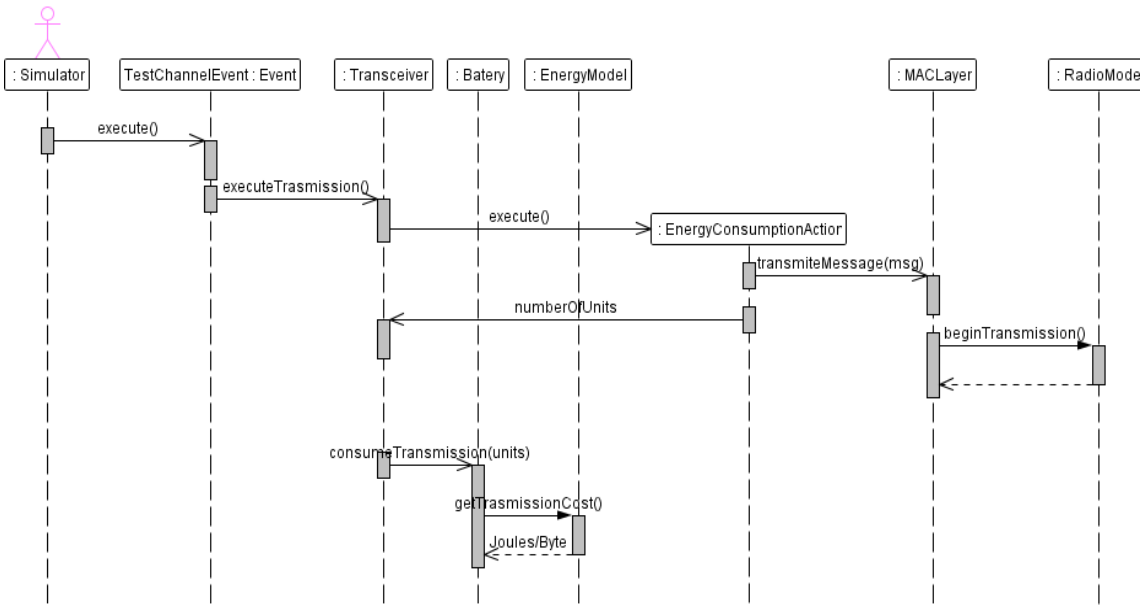

Figura 4.3 Diagrama de sequência para o consumo de energia numa transmissão

Ao nível do componente, no caso o Transciever, para a contabilização do consumo de transmissão, o que se verifica é o seguinte:

```
public void executeTransmission(EnergyConsumptionAction action  ){
  switchON(); \\ transição aleatória de estado
  action.execute(); \\ execução da acção de consumo
  int rate= action.getNumberOfUnits(); \\ contabilização das unidades
  \\ consumo de energia correspondente
  getNode().getBateryEnergy().consumeTransmission(rate);
  switchOFF();
}
```

Ao reutilizar o modelo MAC implementado no simulador JProwler e este não prevendo as mudanças de estado inerentes ao funcionamento de um *mote*, foi implementado um mecanismo aleatório para a transição de estado dos componentes dos sensores. É possível, desta forma, simular a activação (transição do estado desligado para o ligado) de um componente, por exemplo o *Transciever*, para iniciar a transmissão . Assim consegue-se considerar os custos referentes à activação dos componentes depois de estes entrarem em modos de consumo reduzido. O desenho dos sensores organizado por camadas possibilita a implementação de protocolos MAC para diferentes âmbitos, nomeadamente a implementação os protocolos S-MAC, B-MAC, Z-MAC.

4.2.3 Mecanismo de avaliação e testes

O componente de avaliação e execução de testes está ligado intrinsecamente com o simulador. A sua execução resulta, fundamentalmente, da intercepção das acções de envio e recepção de mensagens em cada sensor, ao nível da camada de encaminhamento de dados. No diagrama seguinte procura-se espelhar o funcionamento desta ferramenta relacionado com os acções referidas.

1. Parametrização de um teste

5 . Prova de Conceito da Plataforma

Uma vez implementada a plataforma de simulação é necessário submete-la a uma utilização dirigida para o fim para o qual foi concebida. Tendo isto em atenção, este capítulo apresenta a implementação de dois protocolos de encaminhamento de dados em RSSF. O primeiro trata-se de um protocolo simplista, encaminhamento por *inundação* , que visa, fundamentalmente, demonstrar a facilidade de implementação de um primeiro protocolo e ao mesmo tempo servir de termo de comparação com um protocolo seguro. O segundo protocolo, é um protocolo de encaminhamento seguro de referência, o INSENS[28]. Esta implementação, visa, não só provar que é possível implementar, avaliar e experimentar protocolos de especificação mais complexa, mas também contribuir com uma análise critica deste protocolo em critérios de segurança e no seu impacte nas propriedades que se podem avaliar com recurso à plataforma de simulação.

Com a implementação destes dois protocolos, este capítulo, inicialmente, descreve uma metodologia para a implementação de protocolos de encaminhamento usando a API concebida.

5.1 Metodologia para a implementação de um protocolo

A implementação de um qualquer protocolo pode ser divida em duas componentes: 1) a componente lógica, associada ao funcionamento do protocolo 2) a componente de integração, associada à integração/interacção com a plataforma.

5.1.1 Componente Lógica

Para esta componente é necessário identificar as entidades principais de especificação do protocolo. Entidades estas que serão comuns a qualquer protocolo que se pretenda implementar:

1. Message: Entidade que representa a unidade mínima de envio/recepção de dados. Contém uma estrutura genérica para gestão da informação manipulada, de forma transparente, pelo simulador juntamente com a representação da informação transportada, sob a forma de vector de bytes, assemelhando-se com o formato real de um pacote de comunicação;

2. Node: Esta entidade tem, fundamentalmente, funções associadas a visualização (ex: raio,

cores associadas ao estado) e podendo ter também informação transversal a todas as camadas para facilidade de acesso e que necessita de inicialização prévia (ex. desligado/-ligado), a quando da construção do nó. Poderá não ser implementado se estas condições não se verificarem utilizando-se assim um Node genérico disponibilizado pela API.

3. RoutingLayer: Esta é a entidade principal, uma vez que que contém a lógica do protocolo e que deverá respeitar algumas especificações da API para que a sua integração com o simulador seja adequada;

4. Application: Representa a camada aplicacional que será suportada em termos de encaminhamento pelo protocolo implementado. Pode ser usado para testar comportamentos da mesma aplicação com protocolos de encaminhamento diferentes.

Na figura seguinte observa-se a estrutura do protocolo de inundação criada com base na lógica inerente ao protocolo.

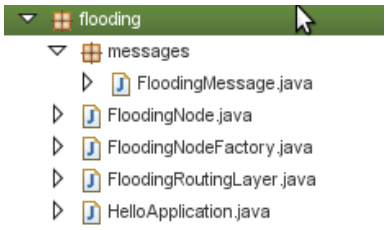

Figura 5.1 Estrutura lógica de um protocolo

5.1.2 Componente de integração

Nesta componente é necessário criar uma entidade que integra/define quais as camadas existentes num sensor genérico de um protocolo. Esta entidade é definida usando um padrão de *factory*. Padrão este, que é consumido pela plataforma para criação de sensores a quando da sua distribuição na área de trabalho. Esta entidade durante a criação dos nós será ainda complementada com informação referente a parâmetros do modelo de energia. Na figura seguinte pode-se observar a simplicidade desta entidade e a forma de integração.

```
public class FloodingNodeFactory extends AbstractNodeFactory {

    public void setup() {
        /* Application to execute */
        setApplicationClass(HelloApplication.class);
        /* Routing protocol*/
        setRoutingLayerClass(FloodingRoutingLayer.class);
        /* Specific Node implementation*/
        setNodeClass(FloodingNode.class);
        /* MAC Layer */
        setMacLayer(Mica2MACLayer.class);
        /* Setup is done */
        setSetup(true);
    }
}
```

Figura 5.2 Definição de uma NodeFactory específica

Quanto à integração com a plataforma, é necessário, ao nível do RoutingLayer, respeitar as especificações definidas pela API do simulador. Salienta-se de seguida as que se julgam essenciais:

5.2 Implementação de protocolo de encaminhamento por inundação

Como já foi referido para efeito de comparação de algumas das propriedades em avaliação foi desenvolvido um primeiro protocolo mais simples que implementa o encaminhamento por inundação (*flooding*). Este protocolo não tem grande complexidade de funcionamento, uma vez que a única restrição para a propagação de mensagens é a garantia de não repetição de mensagens.

5.3 Implementação de Protocolo de Encaminhamento Seguro em RSSF

5.3.1 Fase de desenho dos algoritmos baseado nas especificações

No início desta fase, foi necessário re-aprofundar o funcionamento do algoritmo a implementar, conhecendo e identificando cada mecanismo/técnica especificados, de modo a que se pudesse, dentro do possível, generalizar operações ou interfaces, com vista à reutilização para outros algoritmos, contribuindo para a API genérica de implementação de protocolos de encaminhamento.

6. Avaliação

6.1 Metodologia

6.1.1 Teste A

6.1.1.0.1 O que se quer testar

6.1.1.0.2 Como estou a testar - descrição do setting

6.1.1.0.3 Porque é assim que se avalia

6.1.1.0.4 Quais os resultados

6.1.1.0.5 Conclusões dos testes

7. Conclusões e trabalho futuro

Bibliografia

[1] BTnodes - a distributed environment for prototyping ad hoc networks. http://www.btnode.ethz.ch/Documentation/BTnodeRev3HardwareReference.

[2] The contiki operating system - home. http://www.sics.se/contiki/.

[3] Gateways - crossbow technology. http://www.xbow.com/Products/productdetails.aspx?sid=159.

[4] Home | freemote emulator | assembla. http://www.assembla.com/wiki/show/freemote.

[5] Home (J-Sim official). http://sites.google.com/site/jsimofficial/.

[6] ISIS - JProwler. http://w3.isis.vanderbilt.edu/projects/nest/jprowler/.

[7] ISIS - prowler. http://w3.isis.vanderbilt.edu/projects/nest/Prowler/index.html.

[8] Mica2 Datasheet. http://www.xbow.com/Products/productdetails.aspx?sid=174.

[9] MicaZ Product Details. http://www.xbow.com/Products/productdetails.aspx?sid=164.

[10] Nano-RK: a wireless sensor networking Real-Time operating system. http://www.nanork.org/.

[11] The network simulator - ns-2. http://www.isi.edu/nsnam/ns/.

[12] SENSE-3.0: sensor network simulator and emulator. http://www.ita.cs.rpi.edu/sense/index.html.

[13] ShoX project page. http://shox.sourceforge.net/.

[14] SunSPOTWorld - documentation. http://sunspotworld.com/docs/index.html.

[15] TinyOS community forum || an open-source OS for the networkedsensor regime. http://www.tinyos.net/.

[16] ZigBee alliance. http://www.zigbee.org/Default.aspx.

[17] IEEE Standard for Information Technology - Telecommunications andinformation exchange between systems - Local and metropolitan area networks -specific requirement Part 15.4: Wireless Medium Access Control (MAC) andPhysical Layer (PHY) Specifications fo, 2007.

[18] Kemal Akkaya and Mohamed Younis. A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3(3):325–349, May 2005.

[19] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, March 2002.

[20] J.N. Al-Karaki and A.E. Kamal. Routing techniques in wireless sensor networks: a survey. *Wireless Communications, IEEE*, 11(6):6–28, 2004.

[21] Mohammad Al-Shurman, Seong-Moo Yoo, and Seungjin Park. Black hole attack in mobile ad hoc networks. In *ACM-SE 42: Proceedings of the 42nd annual Southeast regional conference*, pages 96–97, New York, NY, USA, 2004. ACM.

[22] Paolo Baronti, Prashant Pillai, Vince W.C. Chook, Stefano Chessa, Alberto Gotta, and Y. Fun Hu. Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBeestandards. *Computer Communications*, 30(7):1655–1695, May 2007.

[23] M. Blum, Tian He, Sang Son, and John A Stankovic. IGF: a State-Free robust communication protocol for wireless sensor networks.

[24] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance. In *OSDI '99: Proceedings of the third symposium on Operating systems design andimplementation*, pages 173–186, Berkeley, CA, USA, 1999. USENIX Association.

[25] S Corson and J Macker. Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations, January 1999.

[26] George F. Coulouris and Jean Dollimore. *Distributed systems: concepts and design*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1988.

[27] I. Demirkol, C. Ersoy, and F. Alagoz. Mac protocols for wireless sensor networks: a survey. *Communications Magazine, IEEE*, 44(4):115–121, April 2006.

[28] Jing Deng, Richard Han, and Shivakant Mishra. Insens: Intrusion-tolerant routing for wireless sensor networks. *Comput. Commun.*, 29(2):216–230, 2006.

[29] D. Dolev and A. Yao. On the security of public key protocols. *Information Theory, IEEE Transactions on*, 29(2):198–208, 1983.

[30] John Douceur and Judith S Donath. The Sybil Attack. pages 251—-260, 2002.

[31] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on Computer andcommunications security*, pages 41–47, Washington, DC, USA, 2002. ACM.

[32] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. *Hawaii International Conference on System Sciences*, 8:8020, 2000.

[33] Hongmei Deng, Wei Li, and D.P. Agrawal. Routing security in wireless ad hoc networks. *Communications Magazine, IEEE*, 40(10):70–75, 2002.

[34] Fei Hu and Neeraj K. Sharma. Security considerations in ad hoc sensor networks. *Ad Hoc Networks*, 3(1):69–89, January 2005.

[35] Y.-C. Hu, A. Perrig, and D.B. Johnson. Packet leashes: a defense against wormhole attacks in wireless networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer andCommunications. IEEE Societies*, volume 3, pages 1976–1986 vol.3, March-3 April 2003.

[36] Yih-Chun Hu, A. Perrig, and D.B. Johnson. Wormhole attacks in wireless networks. *Selected Areas in Communications, IEEE Journal on*, 24(2):370–380, Feb. 2006.

[37] Yih-Chun Hu and Adrian Perrig. A Survey of Secure Wireless Ad Hoc Routing. *IEEE Security and Privacy*, 2(3), 2004.

[38] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Rushing attacks and defense in wireless ad hoc network routing protocols. In *WiSe '03: Proceedings of the 2nd ACM workshop on Wireless security*, pages 30–40, New York, NY, USA, 2003. ACM.

[39] Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, and Fabio Heidemann, Johnand Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. Netw.*, 11(1):2–16, 2003.

[40] ITU-T. Recommendation X.800: Security Architecture for Open Systems for CCITTApplications, 1991.

[41] David B. Johnson and David A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[42] Holger Karl and Andreas Willig. *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons, 2005.

[43] Chris Karlof and David Wagner. Secure routing in wireless sensor networks: attacks andcountermeasures. *Ad Hoc Networks*, 1(2-3):293–315, September 2003.

[44] Chris Karlof, David Wagner, and Naveen Sastry. TinySec: a link layer security architecture for wireless sensornetworks. pages 162–175, Baltimore, MD, USA, 2004. ACM.

[45] Mauri Kuorilehto and Timo D. Hännikäinen, Marko andHämäläinen. A survey of application distribution in wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2005(5), 2005.

[46] Johannes Lessmann, Peter Janacik, Lazar Lachev, and Dalimir Orfanus. Comparative study of wireless network simulators. In *ICN '08: Proceedings of the Seventh International Conference on Networking*, pages 517–523, Washington, DC, USA, 2008. IEEE Computer Society.

[47] Mark Luk, Adrian Perrig, Ghita Mezzour, and Virgil Gligor. MiniSec: a secure sensor network communication architecture. pages 479–488, Cambridge, Massachusetts, USA, 2007. ACM.

[48] Mark Luk, Adrian Perrig, and Bram Whillock. Seven cardinal properties of sensor network broadcast authentication. pages 147–156, Alexandria, Virginia, USA, 2006. ACM.

[49] Ruiping Ma, Liudong Xing, and H.E. Michel. Fault-intrusion tolerant techniques in wireless sensor networks. In *Dependable, Autonomic and Secure Computing, 2nd IEEE International Symposium on*, pages 85–94, 29 2006-Oct. 1 2006.

[50] Alan Mainwaring, David Culler, Joseph Polastre, and John Szewczyk, Robert andAnderson. Wireless sensor networks for habitat monitoring. In *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensornetworks and applications*, pages 88–97, New York, NY, USA, 2002. ACM.

[51] Aleksandar Milenković, Chris Otto, and Emil Jovanov. Wireless sensor networks for personal health monitoring: Issues and an implementation. *Comput. Commun.*, 29(13-14):2521–2533, 2006.

[52] Wireless Networks. Security Vulnerabilities In Wireless Sensor Networks: A Survey. *Journal of Information Assurance and Security*, 5(2010):031–044, 2009.

[53] J. Newsome, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: analysis & defenses. In *Information Processing in Sensor Networks, 2004. IPSN 2004. Third International Symposiumon*, pages 259–268, April 2004.

[54] E.C.H. Ngai, Jiangchuan Liu, and M.R. Lyu. On the intruder detection for sinkhole attack in wireless sensor networks. In *Communications, 2006. ICC '06. IEEE International Conference on*, volume 8, pages 3383–3389, June 2006.

[55] Sung Park, Andreas Savvides, and Mani B. Srivastava. Simulating networks of wireless sensors. In *Proceedings of the 33nd conference on Winter simulation*, pages 1330–1338, Arlington, Virginia, 2001. IEEE Computer Society.

[56] B. Parno, A. Perrig, and V. Gligor. Distributed detection of node replication attacks in sensornetworks. pages 49–63, 2005.

[57] Bryan Parno, Mark Luk, Evan Gaustad, and Adrian Perrig. Secure sensor network routing: a clean-slate approach. In *CoNEXT '06: Proceedings of the 2006 ACM CoNEXT conference*, pages 1–13, New York, NY, USA, 2006. ACM.

[58] C.E. Perkins and E.M. Royer. *Ad-hoc on-demand distance vector routing*. IEEE, 1999.

[59] Adrian Perrig and Haowen Chan. Security and Privacy in Sensor Networks.

[60] Adrian Perrig, John Stankovic, and David Wagner. Security in wireless sensor networks. *Commun. ACM*, 47(6):53–57, 2004.

[61] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, and J. D. Tygar. Spins: Security protocols for sensor networks. In *Wireless Networks*, pages 189–199, 2001.

[62] Joseph Polastre, Jason Hill, and David Culler. Versatile low power media access for wireless sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networkedsensor systems*, pages 95–107, New York, NY, USA, 2004. ACM.

[63] RFC2828. Internet Security Glossary, 2000.

[64] E. Shi and A. Perrig. Designing secure sensor networks. *Wireless Communications, IEEE*, 11(6):38–43, Dec. 2004.

[65] William Stallings. *Cryptography and Network Security (4th Edition)*. 2005.

[66] Andrew S. Tanenbaum and Maarten Van Steen. *Distributed Systems: Principles and Paradigms*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[67] Hua-Wen Tsai, Chih-Ping Chu, and Tzung-Shi Chen. Mobile object tracking in wireless sensor networks. *Comput. Commun.*, 30(8):1811–1825, 2007.

[68] Yong Wang, G. Attebury, and B. Ramamurthy. A survey of security issues in wireless sensor networks. *Communications Surveys & Tutorials, IEEE*, 8(2):2–23, Quarter 2006.

[69] B.A. Warneke and K.S.J. Pister. Mems for distributed wireless sensor networks. In *Electronics, Circuits and Systems, 2002. 9th International Conference on*, volume 1, pages 291–294 vol.1, 2002.

[70] Anthony D. Wood, Lei Fang, John A. Stankovic, and Tian He. SIGF: a family of configurable, secure routing protocols for wirelesssensor networks. pages 35–48, Alexandria, Virginia, USA, 2006. ACM.

[71] Yang Xiao, Hsiao-Hwa Chen, Bo Sun, Ruhai Wang, and Sakshi Sethi. MAC security and security overhead analysis in the IEEE 802.15.4wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.*, 2006(2):81–81, 2006.

[72] Wei Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *Networking, IEEE/ACM Transactions on*, 12(3):493–506, June 2004.

[73] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. *Computer Networks*, 52(12):2292–2330, August 2008.

[74] Hu. Yih-Chun and A. Perrig. A survey of secure wireless ad hoc routing. *Security & Privacy, IEEE*, 2(3):28–39, May-June 2004.

[75] W. You-Chiun and Y Tseng. Attacks and defenses of routing mechanisms in ad hoc and sensor networks. In *Security in Sensor Networks*.