Avaliação de Mecanismos de Segurança em uma Plataforma para Redes de Sensores Sem Fio*

Germano Guimarães[1], Eduardo Souto[1,2], Judith Kelner[1], Djamel Sadok[1]

[1]Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Caixa Postal 50.740-540 – Recife – PE – Brasil

[2]Universidade Federal do Amazonas (UFAM)
Manaus – AM – Brasil

{gfg, ejps, jk, jamel}@cin.ufpe.br

Abstract. *Wireless sensor networks have recently emerged as successful technologies in a number of applications. The need to build security services into them remains however a considerable challenge as the hardware used still shows serious processing and energy limitations. This work evaluates the impact of a number of security mechanisms on sensor nodes and the network as a whole. Hence a number of measurements were undertaken in a real sensor platform in order to establish accurately energy consumption for various encryption algorithms as well when none of these are used. To our knowledge this is an innovative piece of work that shows through both simulation and measurements that encryption algorithms have different impacts on throughput, delay as well as energy consumption.*

Resumo. *Em grande parte das aplicações para Redes de Sensores Sem Fio verifica-se a necessidade de utilizar mecanismos que forneçam segurança aos dados transmitidos. O propósito deste trabalho é avaliar em uma plataforma real de nós sensores o impacto introduzido pela utilização de mecanismos de segurança do ponto de vista da rede como um todo, observando as limitações destes dispositivos em capacidade computacional e energética. Este trabalho mostra através de simulação e medição que diferentes algoritmos de criptografia influenciam de maneiras diferentes aspectos importantes como a vazão, o atraso de mensagens, além do consumo de energia.*

1. Introdução

As Redes de Sensores Sem Fio (RSSF) nos últimos anos têm sido utilizadas por uma larga faixa de aplicações, como monitoramento ambiental, rastreamento de objetos, monitoramento da saúde humana e sistemas militares [1]. Por outro lado, com a disseminação e popularização das RSSF, a proteção da privacidade dos dados se tornou extremamente importante. A utilização de sensores em sistemas críticos como usinas, aeroportos e hospitais, requer mecanismos que garantam a autenticidade, integridade e confidencialidade aos dados transmitidos. Nestas redes, isto é especialmente difícil porque os nós sensores têm recursos limitados e não se pode garantir a segurança física

* O presente trabalho foi parcialmente financiado pelo CNPq - Processo 55 2111/02-3 e pela CAPES.

dos mesmos. Portanto, uma questão básica é como satisfazer os requisitos das aplicações de forma segura, considerando as características específicas das redes de sensores.

A maioria das pesquisas em redes de sensores está direcionada ao desenvolvimento de novos protocolos que promovam a utilização eficiente dos recursos, principalmente com relação ao consumo de energia [2]. Embora estes protocolos estejam efetivamente prolongando o tempo de vida das redes de sensores, pouca atenção tem sido dada à segurança. Contudo, em muitas aplicações, os aspectos de segurança são tão importantes quanto o desempenho da rede e o baixo consumo de energia.

Nesse contexto, este artigo pretende avaliar o emprego de mecanismos de segurança em RSSF que possibilitem: (1) validar os dados recebidos de um sensor (é necessário autenticar as fontes de maneira que nós maliciosos não possam injetar dados falsos); (2) verificar a integridade dos dados para detectar modificação nos mesmos; (3) garantir a confidencialidade dos dados através do uso de técnicas de criptografia.

Dessa forma, algumas das principais contribuições deste trabalho são:

- Identificação de algoritmos de criptografia apropriados para RSSFs baseado na literatura existente.
- Implementação e análise de cinco algoritmos de criptografia em uma plataforma real de nós sensores.
- Avaliação do impacto da utilização de mecanismos de criptografia em uma rede de sensores. Esta avaliação será realizada através de medições relativas ao consumo de energia (rádio e CPU) e à memória ocupada pelos algoritmos implementados para uma plataforma real de nós sensores.
- Avaliação e validação dos resultados dos diferentes algoritmos de criptografia usando simulação e medição.

A principal meta deste trabalho é mostrar que, apesar das limitações de recursos impostas pelos nós sensores, é possível utilizar alguns mecanismos de segurança, como criptografia e MAC (*Message Authentication Code*), para tornar a troca de informações mais segura. Além disso, este trabalho dá continuidade à avaliação iniciada em [3] onde foram realizadas medições de consumo de energia em um nó sensor real isoladamente utilizando mecanismos de segurança. Uma versão preliminar resumida dos atuais resultados foi apresentada em [4].

Existem outras questões importantes para segurança em redes de sensores, como proteção física dos dados coletados pelos nós sensores, distribuição de chaves e identificação das vulnerabilidades das classes de aplicação. Contudo, estes tópicos estão fora do escopo deste trabalho.

O restante desse artigo está organizado da seguinte maneira: na segunda seção são apresentados alguns trabalhos relacionados com medição de consumo e análise de mecanismos de segurança para RSSF; a Seção 3 aborda questões de segurança em RSSF; a Seção 4 detalha a plataforma utilizada para as medições; a Seção 5 compara os algoritmos de criptografia selecionados para a avaliação; na Seção 6 são mostrados os resultados em uma avaliação de impacto em uma rede de sensores; e a Seção 7 apresenta as conclusões e propostas de trabalhos futuros.

2. Trabalhos Relacionados

Tendo em vista os requisitos de segurança das aplicações e as restrições de energia de redes de sensores, pesquisas direcionadas ao estudo de mecanismos de segurança e à análise do impacto causado por estes mecanismos no consumo de energia dos nós ainda estão em aberto.

Trabalhos relativos ao consumo de energia normalmente restringem-se a simulações baseadas em modelos de energia [5], criação de mapas de energia através de modelos estatísticos [6] e análises matemáticas da redução no consumo devido à implantação de novos algoritmos [2]. No entanto, são encontrados poucos trabalhos voltados a cenários reais nesta área.

Já na área de segurança em RSSF, encontram-se vários trabalhos que propõem soluções para os mais diversos pontos, como roteamento [7] algoritmos de criptografia e autenticação [8], distribuição de chaves [9] e detecção de intrusão [10].

Um dos trabalhos mais reconhecidos em relação à proposição de mecanismos para segurança é o SPINS (Security Protocols for Sensor Networks) [8], onde são definidos dois protocolos, o SNEP (*Sensor Network Encryption Protocol*) e o µTesla. Estes protocolos possuem a finalidade de garantir a confidencialidade, autenticação e integridade dos dados, permitindo que a estação base e os nós possam se comunicar utilizando um roteamento seguro.

Uma camada de enlace segura projetada para redes de sensores sem fio, denominada TinySec foi introduzida em [14]. Genericamente, trata-se de um pacote de segurança que os desenvolvedores de aplicações podem facilmente integrar com suas aplicações de redes de sensores. Este trabalho foi desenvolvido tendo por base as primitivas de segurança disponibilizadas pelo TinySec.

Exemplos de trabalhos correlatos à economia de energia e ao uso de algoritmos de criptografia e autenticação para RSSF são [11], [12], [13]. Entretanto estes trabalhos, ainda pouco aprofundados, necessitam de detalhes a respeito dos seus procedimentos, e principalmente de uma avaliação do real impacto que tais mecanismos irão causar.

3. Segurança em Redes de Sensores

As redes de sensores, utilizando comunicação sem fio, tornam-se mais vulneráveis a ataques, uma vez que neste tipo de comunicação, o modo de transmissão naturalmente utilizado é *broadcast*. Ao utilizar *broadcast*, a rede fica mais susceptível à ação de intrusos, que podem facilmente escutar, interceptar e alterar os dados que trafegam na rede.

Portanto, os requisitos de segurança são essenciais as RSSFs, uma vez que garantem a privacidade entre os nós que fazem parte da rede, além de impedirem o uso indevido e o acesso não autorizado às informações.

A escolha de quais requisitos utilizar depende do tipo de aplicação que está sendo implementada. Dentre os principais requisitos de segurança utilizados, podem ser citados:

- *Autenticidade* - possibilita que o receptor da mensagem seja capaz de verificar a identidade do emissor, evitando dessa forma que nós intrusos injetem dados maliciosos na rede;

- *Confidencialidade* - assegura que o conteúdo da mensagem seja acessado apenas por nós autorizados;
- *Integridade* - garante que se a mensagem tiver seu conteúdo alterado durante a transmissão, estas alterações poderão ser identificadas pelo receptor.

Para garantir a autenticidade, o mecanismo normalmente utilizado é o cálculo de um MAC. O MAC é uma função que utiliza uma chave secreta e a própria mensagem para calcular um *checksum* criptografado. Este *checksum* será utilizado na autenticação e na verificação da integridade da mensagem.

Para garantir a confidencialidade, algoritmos de criptografia são geralmente utilizados. Na subseção seguinte estão relacionados os algoritmos utilizados neste trabalho.

3.1. Algoritmos de Criptografia

Na criptografia assimétrica, o custo relacionado à geração de chaves públicas e privadas é alto. Dessa forma, por consumir menos recursos, a criptografia simétrica é mais adequada para prover segurança em redes de sensores sem fio, uma vez que estas redes são compostas por dispositivos que possuem limitações em relação ao processamento, capacidade de armazenamento, largura de banda e energia [8], [15].

A seguir são apresentados os algoritmos de criptografia utilizados neste trabalho. A simplicidade, eficiência e a quantidade de memória exigida foram os principais critérios utilizados para a seleção destes algoritmos.

De acordo com [8], o RC5 [16] representa o algoritmo de criptografia mais adequado para redes de sensores, devido ao seu alto desempenho e por requerer pouco espaço em memória. Além de ser um algoritmo bastante difundindo e não possuir falhas conhecidas. Em termo de flexibilidade o RC5 é totalmente parametrizado no tamanho de palavra, número de rodadas e tamanho da chave.

O RC6 [17] é um algoritmo que é simples o suficiente para memorizar e pode ser facilmente implementado de forma compacta tanto em software como em hardware. O RC6 não usa tabelas de substituição; ao invés, o principal mecanismo para sua segurança é a técnica de rotacionar dígitos por um número variante de lugares que é determinado pelos dados. Em [18] encontra-se uma aplicação do RC6 para RSSFs.

O TEA [15] (*Tiny Encryption Algorithm*) foi projetado para ser um algoritmo simples que requer pouco espaço em memória. O algoritmo usa uma seqüência de operações sobre palavras ao invés de desperdiçar energia do hardware com operações sobre *byte* ou 4 *bits*. A segurança provida por ele está relacionada ao grande número de iterações utilizadas e não à sua complexidade.

O SkipJack [19] foi projetado para ser usado em *chips* e requer pouco espaço em memória, o que o torna um possível candidato a fornecer segurança em redes de sensores. O SkipJack transforma um bloco de entrada de 64 *bits* dentro de um bloco de saída de 64 *bits*. A transformação é parametrizada por uma chave de 80 *bits* e envolve a execução de 32 iterações de uma função não linear.

O DES (*Data Encryption Standard*) [20] é um algoritmo simétrico que cifra blocos de 64 *bits* utilizando uma chave secreta de mesmo tamanho, onde apenas 56 *bits*

dessa chave são utilizados para cifrar a mensagem, os 8 *bits* restantes servem para calcular a sua integridade, usando o *bit* de paridade. Este algoritmo foi escolhido por ser bastante difundido e por requerer grande capacidade de armazenamento devido às tabelas de consulta utilizadas. Este último fato servirá como base na comparação com os demais algoritmos que requerem pouco espaço de armazenamento.

Existem ainda outros algoritmos interessantes a serem avaliados como Rijndael [21] e TREYFER [22], que exigirem pouca capacidade de armazenamento.

4. Plataforma Utilizada

Nesta seção será apresentada a plataforma de *hardware* e *software* onde foram realizadas as medições de consumo de energia e ocupação de memória, usadas para analisar o impacto da utilização de mecanismos de seguranças em RSSF.

Todas as medições foram realizadas no dispositivo para sensoriamento sem fio MICA2, desenvolvido pela universidade de Berkeley e comercializado pela *Crossbow Technology* [23]. Este dispositivo é composto de um microcontrolador de alta performance e baixa potência, o ATmega128L [24], operando a 7,3728 MHz. Esta plataforma contém 128 KB para memória de programa (FLASH), 4 KB de memória de dados (SRAM) e 512 KB de memória de armazenamento. Para a comunicação sem fio, o MICA2 (Figura 1) utiliza o rádio ChipCon CC1000 [25] com taxa de transmissão de até 38.4 Kbps, freqüência de operação programável entre 433 e 915 MHz e potência de transmissão ajustável entre –20 e +10 dBm.

Figura 1. Foto do sensor sem fio MICA2.

Este nó sensor suporta o sistema operacional de código aberto TinyOS [26] e uma linguagem de programação de alto nível baseada em componentes, chamada nesC [27]. Para prover uma camada de enlace segura no TinyOS foi desenvolvido o TinySec [14], um eficiente cifrador de blocos acoplado à pilha de rádio do sistema operacional. Uma única chave simétrica (distribuída durante a programação dos sensores) é compartilhada com uma coleção de nós da rede. Dessa forma, cada sensor carrega consigo a chave secreta da aplicação. Neste trabalho foi utilizada a versão 1.1.6 do TinyOS que utiliza o protocolo de enlace B-MAC [30].

Antes de transmitir um pacote, cada nó primeiramente cifra os dados utilizando a chave secreta e aplica um código de autenticação da mensagem (MAC) no modo CBC (*Cipher Block Chaining*) para garantir a integridade dos dados. O receptor, através do MAC, verifica se o pacote não foi modificado durante a transmissão pela rede e, em seguida, utiliza a chave secreta de seu conhecimento para decifrar a mensagem.

O TinySec oferece dois modos de segurança: um modo com autenticação, e outro com autenticação e criptografia. Dependendo do modo de segurança, determinado pelo

programador da aplicação, o tamanho do pacote padrão do TinyOS (36 bytes) é alterado, conforme Figura 2 extraída de [14]. No modo apenas com autenticação, o tamanho do pacote tem um acréscimo de 1 byte, já no modo com autenticação e criptografia, este acréscimo é de 5 bytes.

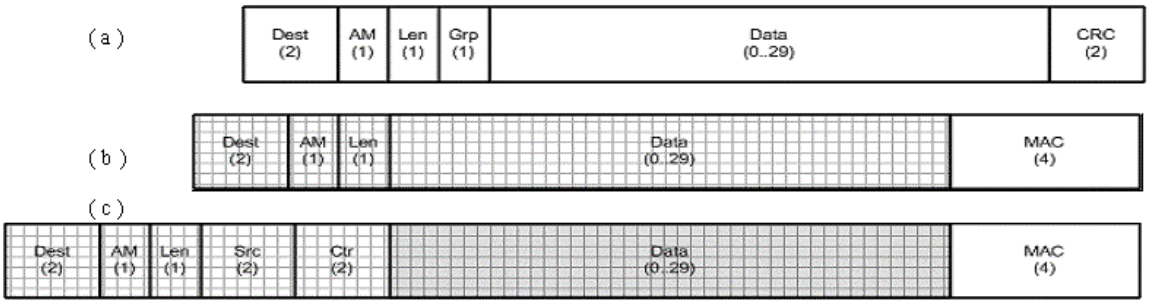

Figura 2. Formato dos pacotes do TinyOS. (a) pacote padrão, (b) pacote com autenticação e (c) pacote com autenticação e criptografia.

No TinySec encontra-se implementado os algoritmos de criptografia SkipJack [19] e RC5 [16]. Neste trabalho outros algoritmos comumente utilizados em sistemas embarcados citados na seção 3.1 foram implementados em nesC e comparados em termos de consumo de energia, processamento e memória.

5. Comparação entre Algoritmos de Criptografia

Em um nó sensor sem fio, com recursos limitados de *hardware* e energia, a utilização de mecanismos de segurança causa um impacto direto no consumo de energia e na ocupação da memória. Os dois dispositivos responsáveis pelo impacto no consumo de energia são a CPU (responsável pelo processamento dos mecanismos de segurança) e o rádio (se a implementação da segurança modificar o tamanho do pacote da rede, como é o caso do TinySec). Quanto à ocupação de memória, o impacto está diretamente relacionado com o acréscimo das linhas de código relativas à implementação dos mecanismos de segurança. Os resultados a seguir foram obtidos utilizando-se a metodologia proposta em [3] que indica um impacto na CPU e na memória para os algoritmos SkipJack, RC5, RC6, TEA e DES.

5.1. Impacto na CPU

O impacto no consumo de energia na CPU, deve-se principalmente ao processamento do algoritmo que irá cifrar o *payload* da mensagem e ao cálculo do código de autenticação deste pacote (cálculo do MAC). Dessa forma, a medição de tempo foi realizada somente durante a execução do procedimento que cifra o *payload* da mensagem e do procedimento que calcula o MAC.

A Tabela 1 exibe o tempo, número de ciclos de CPU e Energia necessários para cifrar o *payload* com 29 bytes e calcular o MAC do pacote pelos algoritmos SkipJack, RC5, RC6, TEA e DES, todos operando no modo CBC [28]. Estes valores representam a média aritmética de 100 medições.

Tabela 1. Impacto na CPU por Pacote

	Cifra do *Payload*			Cálculo do MAC do Pacote		
Algoritmo	*Tempo (ms)*	*Ciclos de CPU*	*Energia (μJ)*	*Tempo (ms)*	*Ciclos de CPU*	*Energia (μJ)*
SkipJack	2,16	15.925,2	51,84	2,99	22.044,6	71,76
RC5	1,50	11.059,2	36,00	2,08	15.335,4	49,92
RC6	10,78	79.478,7	258,72	15,84	116.785,2	380,16
TEA	2,56	18.874,4	61,44	5,07	37.380,1	121,68
DES	608,00	4.482.662,4	14.592,00	1.208,00	8.906.342,4	28.992

Dentre todos algoritmos, o RC5 foi o que apresentou melhor desempenho em termos de tempo (1,50 ms) e por conseqüência, ciclos de CPU (11.059,2 ciclos) e Energia (36,00 microjoules) para cifrar o *payload*. É interessante observar que o desempenho do RC6 foi aproximadamente 7 (sete) vezes pior que o do RC5 em ambos os procedimentos (cifragem do *payload* e cálculo do MAC), apesar de seus autores argumentarem que o RC6 foi projetado para ser mais veloz que o RC5. Resultados semelhantes foram obtidos em [13]. Tanto o RC5 quanto o RC6 foram analisados operando com 12 rodadas.

Como esperado, o DES obteve um desempenho de tempo de processamento bastante inferior a todos os outros, não sendo aconselhada a sua aplicação em nós sensores. Entre os algoritmos avaliados, os melhores resultados foram os obtidos pelos cifradores SkipJack, RC5 e o TEA.

5.2. Ocupação de Memória

O processo de compilação do código fonte nesC para o MICA2 gera um relatório informando em *bytes* a ocupação da memória ROM e RAM. Todas as medições realizadas nesta seção utilizaram uma mesma aplicação, modificando-se apenas os cifradores de blocos. Para efeito de comparação, também foi coletada a ocupação de memória desta aplicação sem utilizar a camada de segurança. Os resultados são apresentados na Figura 3.

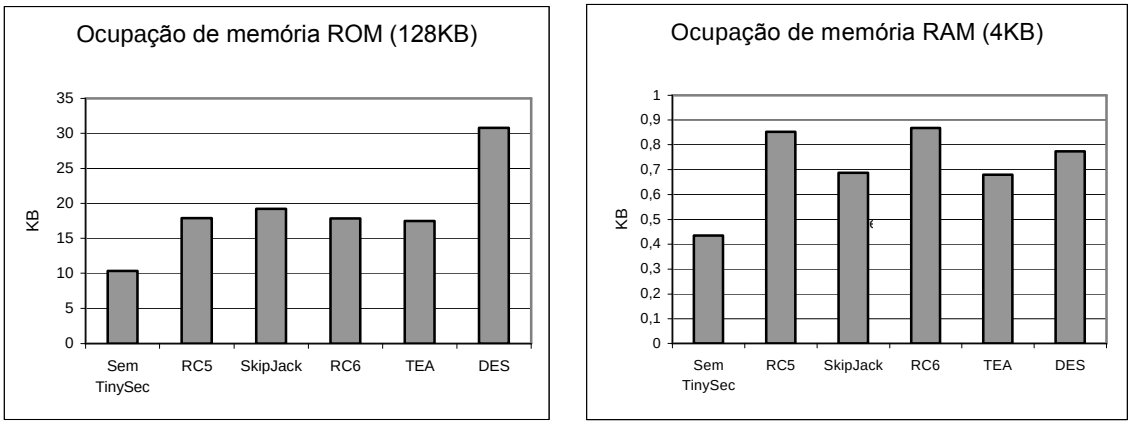

Figura 3. Ocupação de memória ROM e RAM no MICA2 para os algoritmos RC5, SkipJack, RC6, TEA e DES.

Como observado na Figura 3, o acréscimo em ocupação de memória ROM dos algoritmos avaliados ficou entre 7 KB para o TEA e 9 KB para o SkipJack quando comparado com a aplicação sem a camada de segurança TinySec. Valores aceitáveis para a plataforma utilizada nas medições (no caso, o MICA2 com 128 KB de memória ROM). A exceção novamente foi o algoritmo DES cujo acréscimo foi de 20 KB.

Quanto à ocupação de memória RAM, não foram encontrados resultados contrastantes entre os algoritmos. Os melhores resultados foram obtidos pelo TEA (0,68 KB) e SkipJack (0,69 KB), correspondendo a 17 e 17,5% da RAM respectivamente. A aplicação sem a camada de segurança ocupa 0,43 KB, que equivale a 10,75% da memória RAM. Portanto, o custo adicional para a aplicação fazer uso do TEA foi de 6,25%. Novamente, valores aceitáveis para a plataforma MICA2 com 4 KB de memória RAM.

6. Avaliação do Impacto na Rede

Nesta seção serão apresentadas as métricas e os resultados referentes ao impacto da utilização de uma camada de segurança em redes de sensores. As principais variáveis responsáveis por este impacto são o acréscimo em número de bytes no pacote (como apresentado na Figura 2) e o tempo de processamento. As seguintes métricas foram observadas:

- *Vazão de mensagens* - o número de mensagens que chegam ao nó sorvedouro por minuto;
- *Atraso* - a latência entre a origem e o destino da mensagem de acordo com o número de saltos necessário para alcançar o nó sorvedouro;
- *Consumo de energia* - o consumo estimado de energia para uma rede utilizando algoritmos de criptografia com 10, 50 e 100 nós durante 1000 segundos.

Para esta avaliação foram selecionados os três algoritmos que apresentaram melhor desempenho na seção 5: TEA, RC5 e SkipJack. O RC6 que apresentou um desempenho bastante inferior em sua implementação em software com relação ao RC5 foi descartado juntamente com o DES que apresentou um desempenho inviável para nós de sensores.

6.1. Vazão

O cenário de medição é formado por uma rede com 4 nós sensores MICA2, um deles atuando como nó sorvedouro ligado a um computador. Para avaliar esta métrica foi implementada uma aplicação onde os nós simultaneamente enviam mensagens com tamanho máximo permitido (29 bytes) para o nó sorvedouro no tempo mais curto possível. Isto significa que uma nova mensagem é gerada assim que o rádio torna-se disponível. Uma outra aplicação executando no computador contabiliza o número de mensagens recebidas por minuto. Os dados exibidos na Figura 4 representam a média aritmética de 30 medições da quantidade de mensagens recebidas com sucesso pelo nó sorvedouro.

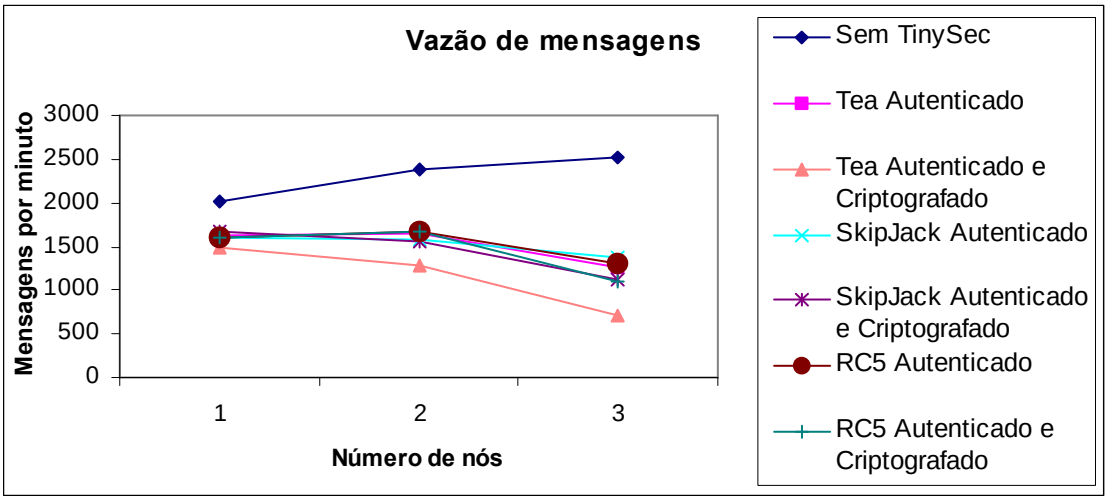

Figura 4. Vazão de mensagens utilizando o TEA, SkipJack e RC5

O gráfico da Figura 4 representa a vazão da rede utilizando os algoritmos TEA, SkipJack e RC5 sob os dois modos de segurança (somente com autenticação, e com autenticação e criptografia). Como pode ser observado e esperado, a melhor vazão foi obtida pela aplicação sem a camada de segurança com a média de transmissão de 2024 mensagens por minuto. Já os valores obtidos com a utilização dos algoritmos de criptografia e MAC estão bastante próximos. Por exemplo, o SkipJack e o RC5 com autenticação obtiveram vazões de 1364 e 1310 mensagens por minuto respectivamente. O pior caso foi constatado para o algoritmo TEA no modo autenticado e criptografado com 719 mensagens por minuto.

É importante ressaltar que ocorreu uma redução na quantidade de mensagens recebidas com sucesso à medida que os nós foram inseridos na rede. São dois os fatores que ocasionam esta redução: o acréscimo em *bytes* no tamanho do pacote, ocasionando uma maior contenção no canal de comunicação, e o *overhead* gerado pelo processamento dos mecanismos de segurança.

6.2. Atraso

O cenário de medição adotado é formado por uma rede composta por 4 nós sensores MICA2, um deles atuando como nó sorvedouro ligado a um computador. Para avaliar esta métrica foi implementada uma aplicação no computador que periodicamente envia mensagens no tamanho máximo permitido (29 bytes) para o nó sorvedouro, o qual é responsável pela difusão dessas mensagens na rede. A idéia é medir o tempo necessário para uma mensagem sair da estação base (nó sorvedouro), viajar pela rede e depois retornar à estação base. Este procedimento também é conhecido como RTT (*Round Trip Time*).

As medições apresentadas na Figura 5 representam a média aritmética de 30 medições do tempo de viagem de um pacote para um número diferente de saltos. Devido à limitação de nós disponíveis durante os experimentos, a mensagem viaja na rede seguindo uma topologia em anel até ter o seu tempo de vida esgotado. Neste momento, a aplicação do usuário calcula o tempo de permanência da mensagem na rede. Este processo é automático e toda vez que uma rodada é concluída, uma nova mensagem com

um tempo de vida maior é enviada à rede e uma nova marcação é realizada. O diâmetro total da rede é de 24 saltos.

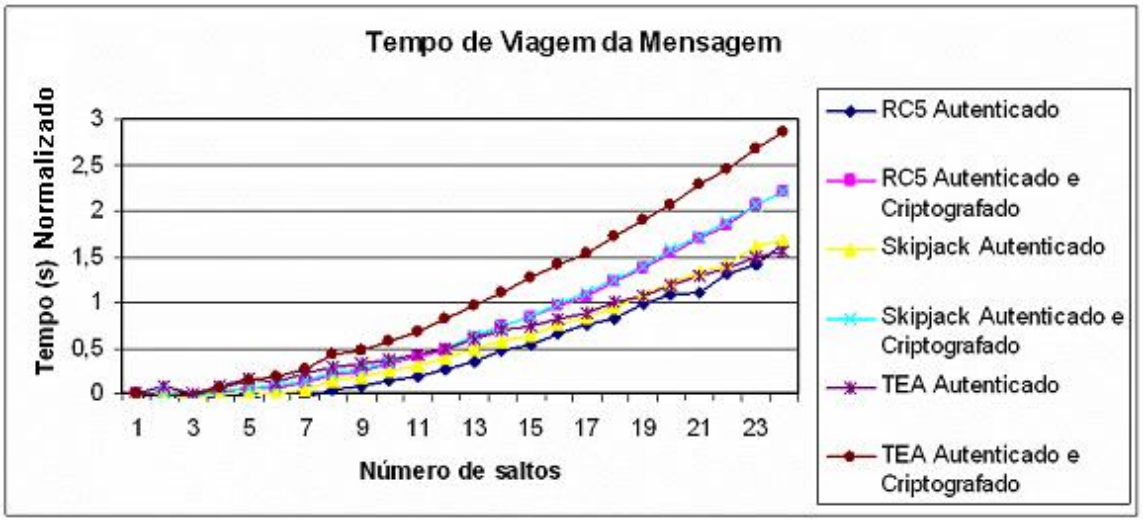

Figura 5. Tempo de viagens de mensagens de acordo com o número de saltos e algoritmo utilizado.

O gráfico da Figura 5 representa o tempo de viagem da mensagem utilizando os algoritmos TEA, SkipJack e RC5 sob os dois modos de segurança (somente com autenticação e com autenticação e criptografia). Estes valores estão normalizados pelos valores obtidos pela aplicação sem a camada de segurança que gasta 13 segundos para realizar 24 saltos (Figura 6). Os resultados utilizando a camada de segurança para os diferentes algoritmos foram bem próximos, variando entre 14.5 segundos para o TEA em modo autenticado e 15.8 segundos para o TEA autenticado e criptografado. Pode-se observar que o acréscimo de tempo entre a aplicação sem e com camada de segurança é inferior a 3 segundos para uma rede com 24 saltos. Essa diferença se torna menor à medida que o número de saltos diminui. Isto prova que para as aplicações sem rígidas restrições de atraso ou redes com pequenos diâmetros, o uso de algoritmos de criptografia é viável.

6.3. Consumo de Energia

Os resultados desta seção foram obtidos através do simulador TOSSIM [26], utilizando o módulo para cálculo do consumo de energia PowerTOSSIM [29]. Este módulo fornece um consumo de energia estimado por dispositivo em cada nó com uma precisão entre 0.45 e 13% para medições em nós reais. Os parâmetros de simulação foram ajustados para corresponder a uma rede real baseada no nó sensor Mica2 [23]. Foi necessária a adaptação do PowerTOSSIM para permitir esta avaliação já que não existe suporte em sua pilha de rádio para TinySec. O modelo de comunicação *multihop* foi adotado entre os nós sensores. Uma topologia em árvore binária foi utilizada para isolar as questões relativas ao roteamento de mensagens na rede.

Foram realizadas simulações em uma RSSF composta por 10, 50 e 100 nós estacionários e homogêneos, distribuídos em uma área de 100x100m^2. Os experimentos foram conduzidos com apenas um nó sorvedouro. O tempo total de simulação foi 1000s. Para avaliar o consumo de energia foi implementada uma aplicação onde os nós

simultaneamente enviam mensagens com tamanho máximo permitido (29 bytes) para o nó sorvedouro a cada 500 ms. Os resultados das simulações são apresentados na Figura 6.

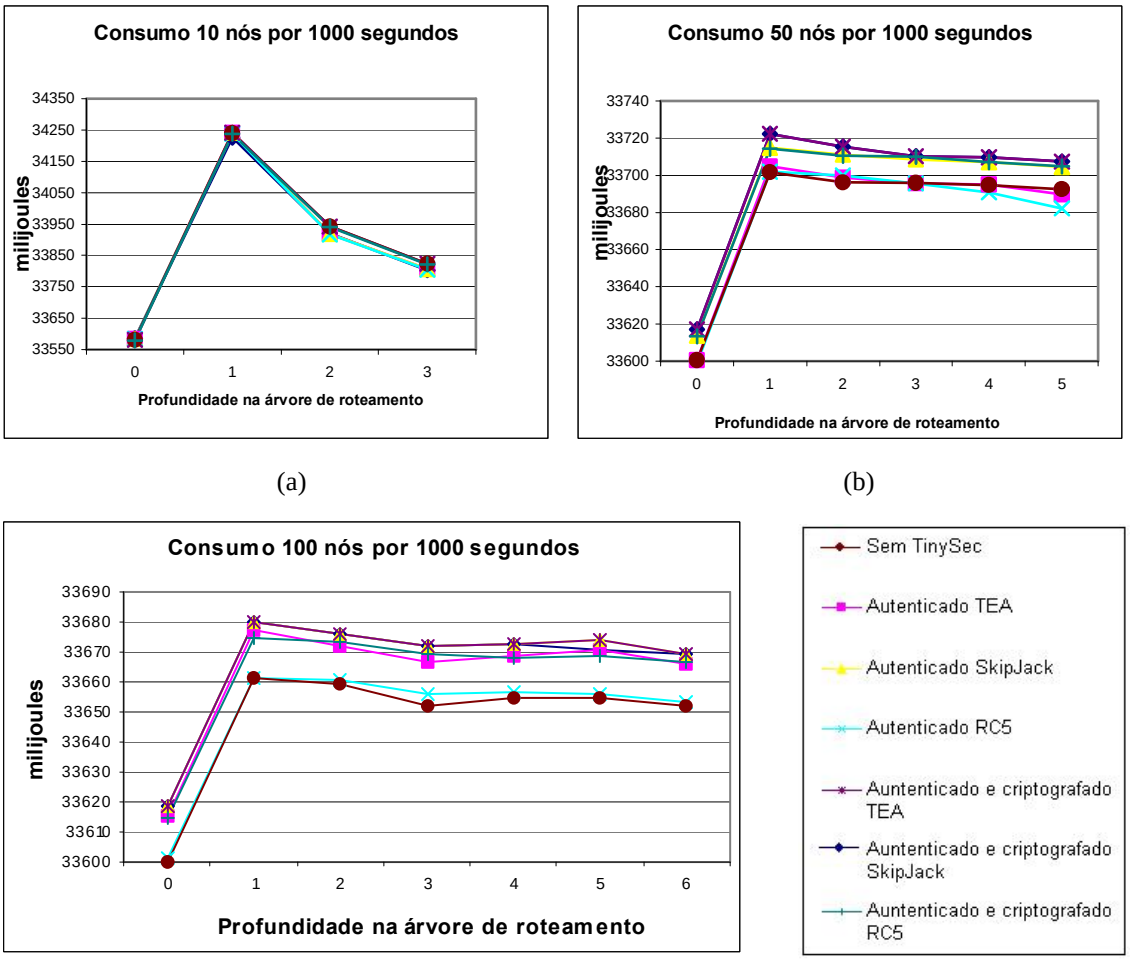

Figura 6. Consumo para uma rede com 10 nós (a), 50 nós (b) e 100 nós (c) após 1000 segundos.

Os gráficos da Figura 6 representam a média do consumo de energia em milijoules dos nós para cada altura da árvore binária. Por exemplo, na altura 0 (zero) encontra-se somente a estação base (nó sorvedouro) e na altura 1 (um) dois nós conectados a estação base (altura zero). A estação base, por não utilizar o rádio para transmitir mensagens, apresentou em todas simulações um baixo consumo. Além disso, devido à comunicação *multi-hop*, o consumo de energia tende a ser maior nos nós mais próximo a estação base, uma vez que estes são responsáveis por rotear um maior número de mensagem.

Como pode ser observado na Figura 6(a), que representa a média do consumo de energia em milijoules para uma rede composta por 10 nós, os resultados obtidos utilizando a camada de segurança para os diferentes algoritmos foram bem próximos em todos os níveis da árvore de roteamento, variando entre 34237 milijoules para o RC5 em modo autenticado e 34241 milijoules para o TEA autenticado e criptografado para altura 1 (um) da árvore de roteamento (altura de maior consumo).

A Figura 6(b) exibe uma rede com 50 nós para o mesmo cenário anterior. Os resultados obtidos foram próximos em todos os níveis da árvore de roteamento, entretanto, diferente do cenário anterior, a variação entre o melhor e pior consumo é de 25 mJ. Isto é devido à disputa pelo acesso ao meio que passou a ter um maior número de nós. Este comportamento também pode ser observado na Figura 6(c) que representa uma rede com 100 nós. Neste cenário, por exemplo, obteve-se 33661 milijoules para o RC5 em modo autenticado e 33679 milijoules para o TEA autenticado e criptografado para altura 1 (um) da árvore de roteamento.

Por fim, observou-se um acréscimo inferior a 25 mJ no consumo de energia para as aplicações sem e com a camada de segurança em todos os cenários. Este acréscimo é justificado quando os requisitos da aplicação exigem integridade, confidencialidade e autenticação aos dados que trafegam na rede.

7. Conclusões e Trabalhos Futuros

Este trabalho apresentou uma avaliação realizada através de medições relativas ao consumo de energia (rádio e CPU) e à memória ocupada pelos algoritmos de criptografia (RC5, RC6, TEA, SkipJack e DES) implementados para uma plataforma real de nós sensores.

Estas medições indicaram que o tamanho do código adicionado à aplicação utilizando alguns algoritmos de criptografia e MAC é aceitável para um nó sensor com 128 KB de memória ROM e 4KB de RAM (MICA2). Quanto ao acréscimo no consumo de energia de um nó sensor, verificou-se que o custo de processamento, referente à utilização dos algoritmos de criptografia, não causa um impacto representativo, já que o mesmo se encontra na casa de microjoules.

Nas avaliações realizadas para um cenário de rede, os resultados relativos à vazão de pacotes indicaram que o uso de criptografia causa uma pequena redução na taxa de entrega de pacotes. Dependendo dos requisitos da aplicação implementada na rede, esta queda pode ser ou não crítica, uma vez que a geração de mensagens em uma rede de sensores pode ser otimizada para evitar gastos de energia desnecessários.

Nas medições referentes ao atraso da entrega de mensagens mostraram que o acréscimo de tempo entre a aplicação sem e com camada de segurança é inferior a 3 segundos para uma rede com 24 saltos. Essa diferença se torna menor à medida que o número de saltos diminui. Isto prova que para as aplicações sem rígidas restrições de atraso ou redes com pequenos diâmetros, o uso de algoritmos de criptografia é possível.

Por fim, os resultados referentes ao consumo de energia revelaram um acréscimo inferior a 25 mJ no consumo para as aplicações sem e com a camada de segurança para todos os cenários avaliados. Isto mostra que apesar das limitações de recursos impostas pelos nós sensores, é possível utilizar alguns mecanismos de segurança, como criptografia e MAC, para tornar a troca de informações na rede mais segura.

Uma proposta de extensão para este trabalho é a implementação de novos algoritmos de criptografia e suas avaliações em outras plataformas de nós sensores, além da avaliação de outras métricas, ampliando a abrangência destes resultados.

Referências

1. I. F. Akyildiz,W. Su, Y. Sankasubramaniam, and E. Cayirci. "Wireless Sensor Networks: A Survey", *Computer Networks*, 38:393–422, 2002.
2. W. Heinzelman, A. Chandrakasan, H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", in *Proc. 33rd Hawaii International Conference on System Sciences*, Jan. 2000, pp. 1–10.
3. G. Guimarães, C. Silva, E.Souto, R.Gomes, J. Kelner and D. Sadok. "Impacto da Utilização de Mecanismos de Segurança em Nodos Sensores", *Proceedings of the VI Workshop de Comunicação sem Fio e Computação Móvel,* Oct. 2004.
4. G. Guimarães, E.Souto, J. Kelner and D. Sadok. "Impacto da Utilização de Mecanismos de Segurança em Redes de Sensores Sem Fio", *Proceedings of the XXIII Simpósio Brasileiro de Redes de Computadores,* Fortaleza, Mai. 2005.
5. L. C. Zhong, J. M. Rabaey, A. Wolisz, "An integrated data-link energy model for wireless sensor networks", in *Proc. International Conference on Communications*, Jun. 2004.
6. R. A. F. Mini, A. A. F. Loureiro, B. Nath, "Prediction-based energy map for wireless sensor networks", in *Proc. 8th IFIP Personal Wireless Communications*, pp. 12-26, Set. 2003.
7. C. Karlof, D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures", in *Proc. 1st IEEE International Workshop on Sensor Network Protocols and Applications*, Mai. 2003.
8. A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen, D. E. Culler, "SPINs: Security Protocols for Sensor Networks", in *Proc. 7th Annual International Conference on Mobile Computing and Networks*, Jul. 2001.
9. D. Liu, P. Ning, "Establishing Pairwise Keys in Distributed Sensor Networks", in *Proc. 10th ACM Conf. on Computer and Communication Security*, Out. 2003.
10. P. Silva, F. Teixeira, W. Chi, José Marcos Nogueira, "Aspectos de Detecção de Intrusos em Redes de Sensores sem Fio", in *Proc. 22nd Simpósio Brasileiro de Redes de Computadores*, Mai. 2004.
11. P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller, M. Sichitiu, "Analyzing and Modelling Encryption Overhead for sensor Network Nodes", in *Proc. 2nd ACM International Conference on Wireless Sensor Networks and Applications*, Set. 2003, pp. 151–159
12. D. Malan, "Crypto for Tiny Objects", Harvard University, Tech. Rep. TR–04–04, Jan. 2004.
13. Y. W. Law, J. Doumen, P. Hartel, "Survey and Benchmark of Block Ciphers for Wireless Sensor Networks", in *Proc. 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, Oct 2004, *to appear*.
14. Karlof. C, Sastry. N, Wagner. David, "TinySec: A Link Layer Security Architecture for Wireless Sensor Networks**",** *Proceedings of ACM SenSys* 2004, November 2004.
15. S. Liu, O. V. Gavrylyako, P. G. Bradford, "Implementing the TEA algorithm on Sensors", in *Proc. 42nd Annual Southeast Regional Conference*, 2004, pp. 64–69.

16. R. L. Rivest, "The RC5 Encryption Algorithm", in *Proc. 1994 Leuven Workshop on Fast Software Encryption*, 1995, pp. 86–96.
17. Rivest, R., Robshaw, M., Sidney, R., Yin, Y., "The RC6 Block Cipher" Specification version 1.1 (1998)
18. Sasha Slijepcevic, Miodrag Potkonjak, Vlasios Tsiatsis, Scott Zimbeck, Mani B. Srivastava, "On Communication Security in Wireless Ad-Hoc Sensor Networks", *Proceedings of the 11th IEEE International Workshops on Enabling Technologies: infrastructure for Collaborative Enterprises*, June 10-12, 2002
19. "SkipJack and KEA algorithm Specifications". *National Institute of Standards and Technology*, Mai. 1998.
20. "Data Encryption Standard (DES)", *National Institute of Standards and Technology*, Dez. 1993.
21. J. Daemen, V. Rijmen, "AES Proposal: Rijndael", Mar. 1999.
22. G. Yuval, "Reinventing the Travois: Encryption/MAC in 30 ROM bytes", in *Proc. 4th Workshop on Fast Software Encryption*", 1997.
23. MICA2: Wireless Measurement System. Disponível: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/6020-0042-05_A_MICA2.pdf
24. Processador Atmega128L. Disponível: http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf
25. ChipCon CC1000 radio. Disponível: http://www.chipcon.com/files/CC1000_Data_Sheet_2_1.pdf
26. J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, K. S. J. Pister, "System Architecture directions for networked sensor", in *Proc. 9th International Conference on Architectural Support for Programming Languages and Operating Systems*, Nov. 2000.
27. D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, D. Culler. "The nesC Language: A Holistic Approach to Networked Embedded Systems" in *Proc. Programming Language Design and Implementation*, Jun. 2003.
28. Mihir Bellare, Joe Kilian, Phillip Rogaway. "The Security of the Cipher Block Chaining Message Authentication Code". *Journal of Computer and System Sciences*, vol 3, Dez. 2000, pp. 362–399.
29. Shnayder, V., Hempstead, M., rong Chen, B., Werner-Allen, G., and Welsh, M. "Simulating the power consumption of large-scale sensor network applications." *In ACM SenSys 2004 (Baltimore, MD, USA, Nov. 2004), ACM Press.*
30. J. Polastre, D. Culler. "B-mac: An adaptive csma layer for low-power operation" , *UC Berkeley, Tech. Rep cs294-f03/bmac*, December 2003.