Uma Avaliação de Protocolos de Acordo de Chave Baseados em Sinais Fisiológicos para Redes Corporais sem Fio

Caio V. F. Silva[1], **Samuel Sena**[1], **Kristtopher Kayo Coelho**[1] **e José Augusto M. Nacif**[1]

[1]Instituto de Ciências Exatas e Tecnológicas – Universidade Federal de Viçosa (UFV)

{caio.fernandes, samuel.sena, kristtopher.coelho, jnacif}@ufv.br

Abstract. *Devices on a wireless body area network monitor clinical signs of patients. The importance contained in these data suggests that they should be transmitted and maintained in a private and secure manner. Thus, the importance of adhering to security protocols, such as authentication between network devices, is evident. However, the devices that make up the body network have specific limitations. These limitations can be pointed out as batteries with low energy capacity, in addition to the reduced data transmission capacity, low processing power, and others. Based on the security principles and the specificities of the network, this work aims to present an empirical analysis about the performance and consumption of hardware resources by authentication protocols. In this investigation, methods that use physiological signals to produce the secret keys of each section were considered. Analyzes were carried out about execution time, memory consumption, goodput, in addition to the false acceptance and false rejection metrics about authentication attempts. When exploring the results obtained, the highlight of a protocol on execution time and the need for more accurate synchronization for application in real scenarios is remarkable. However, the lesser memory consumption and greater precision during the authentication process of the rival method.*

Resumo. *Os dispositivos em uma rede corporal sem fio monitoram sinais clínicos de pacientes. A importância contida nestes dados sugerem que devam ser transmitidos e mantidos de forma privada e segura. Deste modo, evidencia-se a importância da adesão de protocolos de segurança, tal como a autenticação entre dispositivos da rede. Entretanto, os dispositivos que compõe a rede corporal possuem limitações específicas. Estas limitações podem ser apontadas como baterias com baixa capacidade energética, além da capacidade de transmissão dos dados reduzida, pouco poder de processamento, e outras. Com base nos princípios de segurança e as especificidades da rede, este trabalho objetiva apresentar uma análise empírica acerca do desempenho e consumo de recursos de hardware por protocolos de autenticação. Nesta investigação, foram considerados métodos que utilizam sinais fisiológicos para produzirem as chaves secretas de cada seção. Foram realizadas análises acerca de tempo de execução, consumo de memória, goodput, além das métricas de falsa aceitação e falsa rejeição sobre tentativas de autenticação. Ao explorar os resultados obtidos é notável o destaque de um protocolo sobre tempo de execução e necessidade de uma sincronização mais apurada para aplicação em cenários reais. Entretanto destaca-se do outro o menor consumo de memória e maior precisão durante o processo de autenticação.*

1. Introdução

A rede mundial de computadores cresce dia a dia. Parte desse crescimento constante deve-se a evolução das Redes de Sensores Sem Fio (*Wireless Sensor Network* - WSN). WSN são utilizadas em diversos domínios, tais como agricultura, urbanização, segurança e inclusive na medicina [Rawat et al. 2014]. No cenário clínico, utiliza-se uma WSN para realizar monitoramento proativo de pacientes, permitindo inclusive, o tratamento a distância de modo contínuo. Os dispositivos de monitoramento de sinais vitais dos pacientes podem ser classificados como implantáveis ou vestíveis. Atuando de forma cooperativa entre si, tais dispositivos compõe um tipo de WSN denominada Rede Corporal Sem Fio (*Wireless Body Area Network* - WBAN).

Os dispositivos em uma rede WBAN monitoram os sinais clínicos de pacientes, coletando dados como pressão arterial, temperatura, eletrocardiograma (ECG), sinais de fotopletismografia (PPG), taxa de glicose no sangue, fluxo sanguíneo, entre outros. Estes dados aferidos são encaminhados para centros avançados de monitoramento com profissionais especializados em cuidados de saúde [Khan and Yuce 2010]. A importância contida nos dados sugerem que devam ser transmitidos e mantidos de forma privada e segura, de modo a inibir ameaças externas, uma vez que essas informações são sensíveis e individuais e quaisquer alterações podem afetar o diagnóstico de doenças e tratamento. Portanto, é necessário utilizar medidas de segurança para que tais dados sejam transportados e manipulados [Maisel 2010].

Com base na necessidade de segurança, protocolos de autenticação de usuários em dispositivos WBAN são indispensáveis. Entretanto, os dispositivos que compõe a WBAN possuem limitações. Estas limitações podem ser exemplificadas por baterias com baixa capacidade energética, além da capacidade de transmissão dos dados reduzida, pouco poder de processamento, entre outras [Cherukuri et al. 2003]. Ademais, uma WBAN deve atuar de forma escalável, resiliente e energeticamente eficiente. Estes requisitos são importantes, principalmente quando conceitua-se aplicações com dispositivos implantáveis. Pois estes devem manter-se em pleno funcionamento o maior tempo possível para que não ocorra desconforto recorrente à remoção, substituição ou recarga de um dispositivo inoperante. Além disso, é interessante que uma WBAN tenha a facilidade para adicionar e configurar novos dispositivos em uma rede (*Plug and Play*) [Ali and Khan 2015]. Portanto, é necessário desenvolver protocolos de segurança que utilizem os recursos de hardware de maneira eficiente atendendo às exigências das WBANs.

É notável que existam vários protocolos para acordos de chaves para WSN propostos na literatura [Nguyen et al. 2015]. Entretanto é necessário uma análise aprofundada destas soluções, de modo que consigam atender as especificidades rede de uma WSN corporal ou WBAN. Os protocolos que empregam sinais fisiológicos para definirem chaves secretas e compartilhá-las entre dispositivos tendem a atender aos requisitos desejáveis em uma WBAN [Ali and Khan 2015]. Os sinais fisiológicos comumente utilizados em protocolos de estabelecimento de chaves envolvem o ECG e PPG [Ali and Khan 2015]. A literatura indica que métodos de acordo de chaves seguem primitivas difusa (*Fuzzy Primitive*) ou a primitivas não difusa (*Non-fuzzy Primitive*). A *Fuzzy Primitive* tem como objetivo esconder um segredo mesclando-o com outros dados. Para isto podem ser utilizadas técnicas de compromisso difuso (*Fuzzy Commitment*) ou cofre difuso (*Fuzzy Vault*) [Marin et al. 2016]. Em métodos conhecidos como *Fuzzy Commit-*

ment, é estabelecida uma chave aleatória, a qual é combinada aos sinais fisiológico do usuário [Juels and Wattenberg 1999]. Para a classe de algoritmos *Fuzzy Vault*, a autenticação é dada pela reconstrução de um polinômio. Tal polinômio é inicialmente elaborado com características de sinais fisiológicos, adicionado de pontos falsos, os quais são produzidos aleatoriamente [Juels and Sudan 2002]. A *Non-fuzzy Primitive* engloba estratégias que não utilizam técnicas para combinar um segredo aleatório, porém ainda utilizam sinais fisiológicos para estabelecer o acordo. Como por exemplo pode-se destacar [Venkatasubramanian et al. 2008], que gera uma chave a partir do próprio sinal fisiológico. [Wang et al. 2011] utiliza o modelo oculto de Markov (*Hidden Markov Model* - HMM). Além de [Wang et al. 2010], que faz uso do modelo de mistura gaussiana (*Gaussian Mixture Model* - GMM).

A principal contribuição deste trabalho é fundamentada por apresentar uma análise de desempenho comparativa entre um método que segue a *Fuzzy Primitive* e outro que segue uma *Non-fuzzy Primitive*. O objetivo é avaliar as duas classes de protocolos que utilizam sinais fisiológicos para o acordo de chaves. Realizamos investigações acerca de tempo de execução, consumo de memória, *goodput* (quantidade de dados úteis transmitidos durante um intervalo), além das métricas de falsa aceitação (*False Acceptance Rate* - FAR) e falsa rejeição (*False Rejection Rate* - FRR) sobre tentativas de autenticação. Ao fim da análise percebeu-se que o protocolo *Fuzzy Primitive* selecionado apresentou um tempo de execução mais elevado, consumiu menos memória e teve um *goodput* ligeiramente inferior se comparado ao protocolo *Non-Fuzzy Primitive*. Além disso, constatou-se que o protocolo *Non-Fuzzy Primitive* selecionado possui uma limitação quanto à sincronização dos dispositivos. Os protocolos implementados para esta avaliação estão disponíveis em repositórios públicos [1] [2], de modo a contribuir com a evolução do tema junto a comunidade científica.

A organização do artigo segue como descrita. A seção 2 aborda uma discussão sobre trabalhos relacionados. A seção 3, detalha os protocolos de autenticação avaliados. A seção 4 descreve a metodologia da avaliação dos protocolos. Na seção 5 apresenta-se os resultados e, por fim, a seção 6 exibe as considerações finais e trabalhos futuros.

2. Trabalhos relacionados

Os dispositivos que compõe uma rede corporal possuem limitações, dentre elas: baixa capacidade de processamento e armazenamento de dados, além de recursos energéticos escassos e taxa de comunicação reduzida [Cherukuri et al. 2003]. Portanto, é imprescindível que os protocolos de acordo de chaves atendam a estas necessidades básicas impostas pelos hardwares dos dispositivos WBAN. Além disso, os esquemas de segurança devem atender à critérios de protocolos biométricos [Poon et al. 2006], tais como distintividade (distinguível entre pessoas diferentes) e invulnerabilidade (resistente a ataques).

Em [Juels and Wattenberg 1999], é apresentado um esquema de acordo de chaves com a proposta de utilizar biometria para autenticação. Esse esquema segue uma *Fuzzy Primitive* em conjunto com códigos de correção de erro e criptografia para garantir o acordo de chaves em segurança, introduzindo o conceito

[1] https://github.com/caiofers/pska2010
[2] https://github.com/caiofers/eka2008

Fuzzy Commitment. Entretanto, a análise de segurança realizada é baseada em dados biométricos como digital de algum dedo e íris. Ademais, o esquema não foi desenvolvido especificamente para redes de dispositivos corporais envolvendo sinais como ECG e PPG. Portanto, surgiram propostas voltadas para WBANs, como [Venkatasubramanian et al. 2009], que apresenta o protocolo *Physiological-Signal-based Key Agreement* (PSKA), e [Venkatasubramanian et al. 2008] que demonstra o protocolo *Electrocardiogram based Key Agreement* (EKA).

O protocolo PSKA segue a *Fuzzy Primitive*, utilizando o conceito de *Fuzzy Vault*, introduzido por [Juels and Sudan 2002]. O protocolo cumpre os critérios de segurança gerando chaves longas e aleatórias, utilizando sinal fisiológico distinguível entre pessoas e que varia com o tempo. Além da análise de segurança, uma análise de desempenho também é realizada, avaliando custo computacional em termos de ciclos de relógio e a quantidade de uso de memória.

O EKA é um protocolo de acordo de chaves que segue a *Non-Fuzzy Primitive* e foi elaborado para aplicações em WBANs. Este protocolo utiliza sinais fisiológicos como ECG para estabelecer um acordo de chaves. Em tal trabalho é realizado uma análise de segurança do esquema proposto. Perante a análise, são apresentados resultados com relação a distintividade entre pessoas, aleatoriedade e variação temporal da chave, uma vez que os sinais fisiológicos variam com o tempo. Por mais que o trabalho tenha realizado a análise de segurança, uma análise de desempenho não foi realizada. Portanto, existe a necessidade de verificar o comportamento do protocolo durante a execução.

Tendo em vista os trabalhos acima listados, o presente artigo objetiva apresentar uma análise comparativa das etapas de cada protocolo acerca de seus respectivos desempenhos. Assim complementando a análise apresentada em [Venkatasubramanian et al. 2009] e [Venkatasubramanian et al. 2008].

3. Protocolos de acordo de chave

Nesta seção os protocolos selecionados para avaliação são descritos, apresentando características intrínsecas sobre cada um deles. Como a natureza da análise é uma comparação de desempenho entre protocolos para WBAN de classes diferentes, os protocolos selecionados atendem de antemão aos requisitos desejáveis das redes corporais listados em [Ali and Khan 2015]. Dentre os requisitos observados estão:

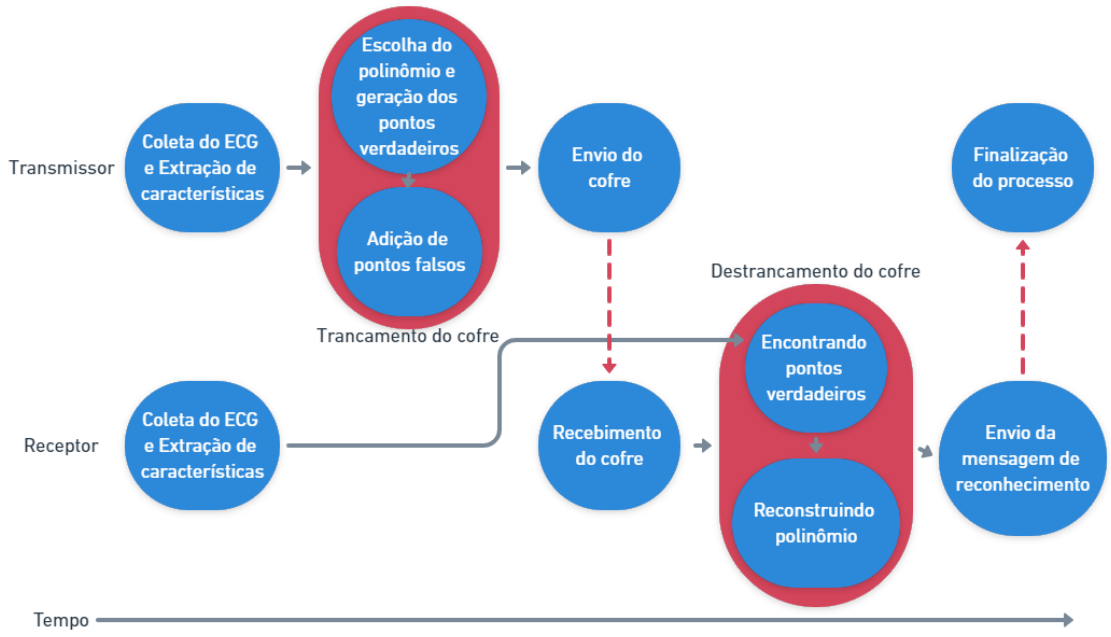
- Escalabilidade: Suficientemente escalável de modo que a segurança não seja comprometida ao adicionar um novo nó na rede;
- Resiliência a ataques: Capacidade de se opor à um ataque que deseja capturar um nó da rede;
- Eficiência energética: Baixo consumo ao gerar as chaves e fazer um acordo de chaves entre os nós;
- *Plug-and-play*: Permite a adição de novos nós que automaticamente começam a gerar chaves e fazer acordos sem que haja ações de terceiros;

Portanto, os trabalhos eleitos para avaliação foram o *Fuzzy Vault* (seção 3.1) apresentado por [Venkatasubramanian et al. 2009] e o *Electrocardiogram Based (EKG-Based)* (seção 3.2) [Venkatasubramanian et al. 2008] após observação de informações levantadas em [Ali and Khan 2015]

3.1. *Fuzzy Vault*

O *Fuzzy Vault* [Juels and Sudan 2002] é uma técnica de acordo de chaves que tranca um "segredo" em um cofre. Tratando-se do PSKA [Venkatasubramanian et al. 2009], em seu cofre são guardados pontos $(v, P(v))$, onde v é um valor que pertence ao vetor de características e $P(v)$ é o resultado do polinômio que utiliza a característica v como variável. Após chegar ao nó de destino, o polinômio é reconstruído utilizando tais pontos a fim de concretizar a autenticação. Na Figura 1 é possível ver o funcionamento em alto nível do protocolo. Em seguida há uma descrição mais detalhada do método.

Figura 1. Protocolo PSKA



3.1.1. Vetor de características

Inicialmente, nós sensores em uma rede WBAN coletam sinais fisiológicos de forma síncrona. Esta coleta é realizada durante um período de tempo definido e com uma taxa de coleta constante, por exemplo 125Hz durante cinco segundos. Em seguida os dados são divididos em cinco partes, denominadas janelas, cada uma com duração de um segundo. A cada janela é aplicada a transformada rápida de Fourier (*Fast Fourier Transform* - FFT) de 128 pontos, a qual transforma o sinal do domínio de tempo, para domínio de frequência. Quando o sinal é transformado pela FFT, ele passa a ter um comportamento simétrico, possibilitando o uso de apenas metade dos pontos, ou seja, 64 pontos.

Após a transformação, um algoritmo de detecção de pico é aplicado aos pontos para extrair tuplas $< m, i >$, onde m é o valor do pico e i é seu respectivo índice. Posteriormente a identificação de todos os picos, ocorre a quantização. Na quantização é realizado um mapeamento, convertendo tanto o valor m como o índice i de cada um dos

picos em um outra tupla com valores menores representando um sinal digital. Em seguida é realizada a conversão de m e i de cada tupla em uma *string* binária. Por fim, ao concatenar $m|i$ obtém-se uma característica v. Ao findar processo de concatenação para todas as tuplas $< m, i >$, o vetor de características V é constituído em cada um dos sensores, denominados V_t, para o sensor transmissor, e V_r, para o sensor receptor.

3.1.2. Geração do cofre

Após produzir o vetor de características V_t, o nó transmissor gera um polinômio de ordem N. Essa ordem é conhecida entre os nós na rede e o polinômio gerado tem o formato da equação 1. Onde cada coeficiente C é gerado aleatoriamente. Os coeficientes são concatenados para, em composição, formarem a chave secreta (equação 2), a qual é utilizada na mensagem de troca entre os nós.

$$p(x) = C^N x^N + C^{N-1} x^{N-1} + \ldots + C^0 \tag{1}$$

$$K = C^N | C^{N-1} | \ldots | C^0 \tag{2}$$

O cofre é composto por um conjunto de tuplas T, e cada tupla é definido por $\{v, p(v)\}$, onde v é uma característica do vetor de características V_t e $p(v)$ é o cálculo do polinômio para v. Além dos pares de tuplas T, também são geradas tuplas aleatórias, chamadas de pontos falsos. Os pontos falsos são representados por $\{f, r\}$, onde f não pertence ao vetor de características V, e r é diferente de $p(f)$. Estes pontos falsos formam o conjunto F.

3.1.3. Trancar o cofre e trocar mensagem

Para trancar o cofre, os conjuntos T e F são permutados aleatoriamente (equação 3), formando o novo conjunto de pontos U. Portanto, os pontos legítimos ficam misturados aos pontos falsos, concluindo o trancamento do cofre. Logo que o cofre é trancado, o mesmo é enviado para o nó receptor por meio da mensagem 4, onde ID_t é o identificador do nó transmissor, ID_r é o identificador do nó receptor, U é o cofre trancado, N_o é um número aleatório para manter a transação recente e o MAC (***Message Authentication Code***) é a mensagem de autenticação , o qual contém a chave trancada no cofre.

$$U = Permut(T \cup F) \tag{3}$$

$$< ID_t, ID_r, U, N_o, MAC(K, U|N_o|ID_t) > \tag{4}$$

3.1.4. Destrancando e confirmando o cofre

Nesta etapa, o dispositivo receptor utilizará o vetor de características V_r produzido por si próprio em conjunto com o cofre U recebido do transmissor. Considerando que cada ponto de U seja representado como (a, b), é realizada uma interseção entre os pontos que tem a pertencente ao vetor de características V_r. Essa operação de interseção acarreta na extração de possíveis pontos verdadeiros do cofre U. Um novo conjunto L, contendo somente os pontos extraídos anteriormente, é criado e utilizado para reconstruir o polinômio por meio de alguma estratégia de interpolação de pontos. Especificamente no caso do PSKA [Venkatasubramanian et al. 2009], é utilizada a interpolação de Lagrange. Para realizar a interpolação, é necessário $n + 1$ pontos de L de modo que a cardinalidade de L seja maior que n. Como resultado da interpolação, obtém-se um polinômio. A partir dos coeficientes deste novo polinômio a chave secreta é reconstruída e utilizada para a verificação do MAC, o que destrancará ou não o cofre.

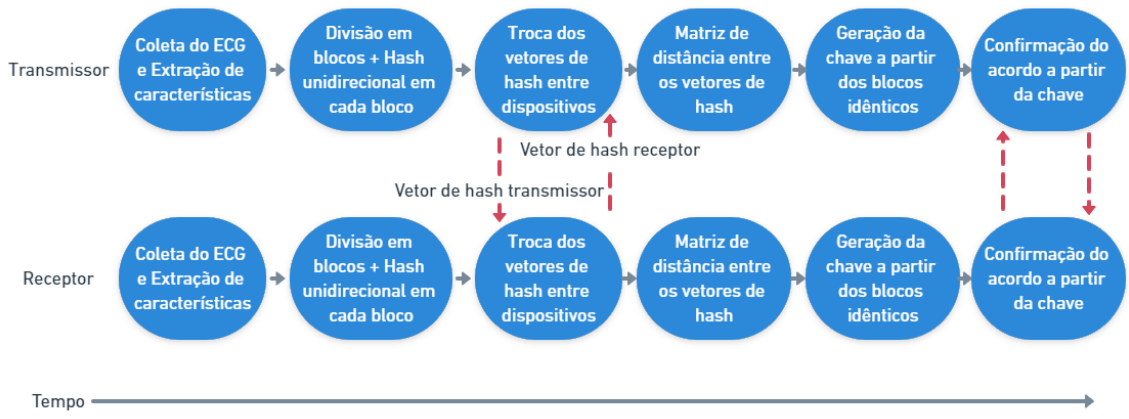
Após destrancar o cofre com sucesso, o nó receptor envia a mensagem 5 com a chave para o nó transmissor, confirmando que o cofre chegou ao receptor. Desta forma a autenticação é confirmada.

$$< MAC(K, N_o|ID_t|ID_r) > \tag{5}$$

3.2. EKG-Based

A técnica *EKG-Based* se diferencia do *Fuzzy Vault* pelo fato da chave comum entre o transmissor e o receptor ser constituído por sinais fisiológicos, especificamente o eletrocardiograma. Tratando-se do EKA [Venkatasubramanian et al. 2008], os dispositivos trocam os vetores de característica entre si e geram uma chave comum a partir de uma matriz de distância entre o vetor do transmissor e o vetor do receptor como pode ser visto na Figura 2.

Figura 2. Protocolo EKA



3.2.1. Vetor de características

A constituição do vetor de características segue etapas semelhantes às descritas em *Fuzzy Vault* do PSKA, até a aplicação da transformada. No EKA ambos os dispositivos (transmissor e receptor) coletam dados a uma taxa fixa de amostras por segundo. As amostras são filtradas, única diferença se comparado ao PSKA, e divididas em 5 janelas. Em cada uma das janelas é aplicado a FFT de 128 pontos, e assim como no PSKA, somente os primeiros 64 destes 128 pontos serão utilizados e denominados coeficientes. Após a aplicação da FFT ocorre a concatenação das janelas com os 64 coeficientes. Deste modo, é produzido o vetor de características V contendo 320 coeficientes. O vetor de características V é posteriormente dividido em 20 blocos contendo 16 coeficientes cada. Cada coeficiente é quantizado (de modo equivalente a [Venkatasubramanian et al. 2009]) e representado por uma *string* binária contendo quatro bits. O resultado final desta operação produz 20 blocos de 64 bits cada.

3.2.2. Fase de compromisso

Após os dispositivos que desejam trocar informações entre si produzirem seus respectivos blocos, é iniciada a fase de comprometimento, ou compromisso. Os blocos B_t do transmissor são representados pela Equação 6 e os blocos B_r do receptor pela Equação 7.

$$B_t = \{b_1^t, b_2^t, \ldots, b_{20}^t\} \tag{6}$$

$$B_r = \{b_1^r, b_2^r, \ldots, b_{20}^r\} \tag{7}$$

Nesta etapa, uma função *hash* unidirecional é aplicada em cada um dos blocos de 64 bits dos dispositivos. Os blocos com o *hash* aplicado, são trocados entre os dispositivos. Ou seja, os blocos do transmissor com o *hash* aplicado são enviados para o receptor (Mensagem 8) e vice-versa (Mensagem 9). Além dos blocos, uma chave aleatória gerada em cada um dos sensores também é enviada, denominadas K^t para o transmissor e K^r para o receptor.

$$\begin{aligned} < ID, N, hash(b_1^t, N), hash(b_1^t, N), \ldots, \\ hash(b_{20}^t, N), MAC(K^t, ID, N, hash(b_1^t, N), hash(b_1^t, N), \ldots, \\ hash(b_{20}^t, N)) > \end{aligned} \tag{8}$$

$$\begin{aligned} < ID, N, hash(b_1^r, N), hash(b_1^r, N), \ldots, \\ hash(b_{20}^r, N), MAC(K^r, ID, N, hash(b_1^r, N), hash(b_1^r, N), \ldots, \\ hash(b_{20}^r, N)) > \end{aligned} \tag{9}$$

3.2.3. Fase de processamento

Nessa fase os blocos que são idênticos em ambos os sensores serão identificados e utilizados para criar uma chave comum. Para fazer a identificação, uma matriz H é calculada com base na distância de *Hamming*. Esse cálculo é realizado entre os *hashs* dos blocos pertencente a B_r, do dispositivo receptor, e os *hashs* dos blocos pertencente a B_t, recebidos do outro dispositivo. Os índices da matriz indicarão quais os blocos que geraram os hashs. Com base nos índices, serão selecionados os blocos que tem o *hash* idêntico, ou seja, distância zero. Tais blocos serão utilizados para construir a chave comum entre os sensores. (Note que são os blocos que serão utilizados e não o *hash* dos blocos). Ao identificar todos os blocos idênticos, estes são agrupados em uma lista e posteriormente é aplicada uma função de *hash* em toda a lista. Assim é produzido a chave comum R para o dispositivo transmissor e R' para o receptor.

3.2.4. Fase de descompromisso

O descompromisso é a fase final em que ocorre a legitimação dos blocos enviados na fase de compromisso. Para que isso aconteça, os dispositivos comunicam entre si enviando uma disjunção exclusiva entre a chave comum e a chave aleatória obtida na fase de compromisso. A Mensagem 10 é enviada do transmissor para o receptor e a Mensagem 11 é enviada do receptor para o transmissor.

$$< X = K^t \oplus R, MAC(R, X) > \tag{10}$$

$$< X' = K^r \oplus R', MAC(R', X) > \tag{11}$$

No momento em que a mensagem chega ao outro dispositivo, o MAC é verificado utilizando a chave comum extraída dele. Se a verificação for um sucesso, o nó vai obter a chave aleatória produzida anteriormente manipulando a chave comum e a operação XOR. Caso a chave estiver correta, a verificação foi concluída e a autenticação realizada.

4. Metodologia

Nesta seção são apresentados os materiais e métodos utilizados para realizar a análise dos protocolos. Inicialmente é revelada a base de dados utilizada e em seguida é exibido os parâmetros dos protocolos, meios de realizar a simulação para a análise empírica e configurações do dispositivo.

4.1. Base de dados

Para realizar a avaliação dos protocolos de acordo de chaves [Venkatasubramanian et al. 2008, Venkatasubramanian et al. 2009], utilizou-se a base de dados *Lobachevsky University Electrocardiography Database* (LUDB) [Kalyakulina et al. 2018]. Essa base é pública, mantida pelo *Massachusetts Institute of Technology* (MIT), e pode ser obtida em Physiobank [3]. A LUDB, contém

[3]https://www.physionet.org/content/ludb/1.0.0/

200 registros de ECG, coletados entre 2017 e 2018. Estes registros são compostos por 10 segundos de aferições à 500 Hz. A idade dos voluntários no banco de exames varia entre 11 e 89 anos, com uma média de 52 anos. A distribuição por gênero é representada por 42,5% mulheres e 57,5% homens. Além disso, os sinais foram coletados de voluntários saudáveis e de pacientes do Hospital Nizhny Novgorod City. Os pacientes manifestavam doenças cardiovasculares, apresentando sinais de corações com taquicardia, bradicardia e arritmia.

4.2. Parâmetros de implementação e validação

Ambos protocolos foram implementados utilizando a linguagem Python 3.8. Para validação das implementações foram realizados testes de taxa de falsa rejeição (FRR) e taxa de falsa aceitação (FAR). Utilizou-se 50 conjuntos de entrada extraídos do LUDB, o suficiente para realizar a validação visto que em [Venkatasubramanian et al. 2008] e em [Venkatasubramanian et al. 2009] foram utilizados uma quantidade de menor de amostras. Para os testes de falsa rejeição foram executados 100 ciclos de acordo de chave válidos. Ao final da execução foi calculado a quantidade de vezes que um acordo foi rejeitado em relação a quantidade de acordos que deveriam ter sido aceitos, resultando na métrica FRR. Para testes de falsa aceitação também foram executados 100 ciclos, porém cada ciclo nesse caso deveria resultar em uma rejeição. Da mesma forma, foram contabilizados a quantidade de vezes que um acordo foi aceito quando na verdade deveria ser rejeitado, chegando à métrica FAR.

Além disso, os protocolos possuem alguns parâmetros de implementação. Os valores foram selecionados visando a reprodução dos resultados obtidos em [Venkatasubramanian et al. 2008] e em [Venkatasubramanian et al. 2009], levando em consideração a FRR e FAR. Um parâmetro presente no dois protocolos é a quantidade de "janelas"(seções) em que as amostras são divididas para extração de características. O número de janelas em ambos protocolos foi definido como oito. No protocolo PSKA, a ordem do polinômio que será utilizada para trancar o cofre foi definida como oito. E no EKA existe a parametrização da quantidade de blocos em que os coeficientes serão subdivididos para executar a fase de compromisso, a qual foi definida como 20 blocos.

4.3. Análise de desempenho empírica

A análise de desempenho foi realizada em ambos protocolos e avaliada em etapas essenciais de cada um deles separadamente, tendo como objetivo proporcionar de forma empírica uma visão do consumo de recursos durante cada fase de execução. A execução dos protocolos foi realizada com 50 registros de sinais de ECG, cada um contendo 10 segundos de amostragem à 500 Hz, resultando em 5000 amostras. Em seguida, foi realizado o cálculo da média e desvio padrão do tempo de execução (em milissegundos) de cada módulo. O consumo de memória RAM também foi computado com execuções múltiplas com o auxilio da biblioteca *tracemalloc* do Python para traçar a quantidade de blocos de memória alocados em cada etapa. Ademais, uma breve analise de *goodput* foi realizada a fim de medir a quantidade de acordos que podem ser computados por segundo. O tamanho das mensagens em bits foi estimado manualmente observando as informações a serem transmitidas e o *bit rate* considerado foi de 11.11 Kbps alcançado em [Vale-Cardoso et al. 2020].

Os testes foram simulados em um Raspberry Pi 3 Model B+ com um processador Broadcom BCM2837B0 64bits ARM Cortex-A53 Quad-Core 1.4GHz e 1 GB de memória RAM. Apesar de não ser um dispositivo WBAN, os resultados obtidos são relevantes visto que os experimentos com os protocolos foram executados em seus estados equivalentes de funcionamento, utilizando o mesmo dispositivo e plataforma, com as mesmas amostras, mesma quantidade de entrada e em um dispositivo com recursos limitados.

5. Resultados

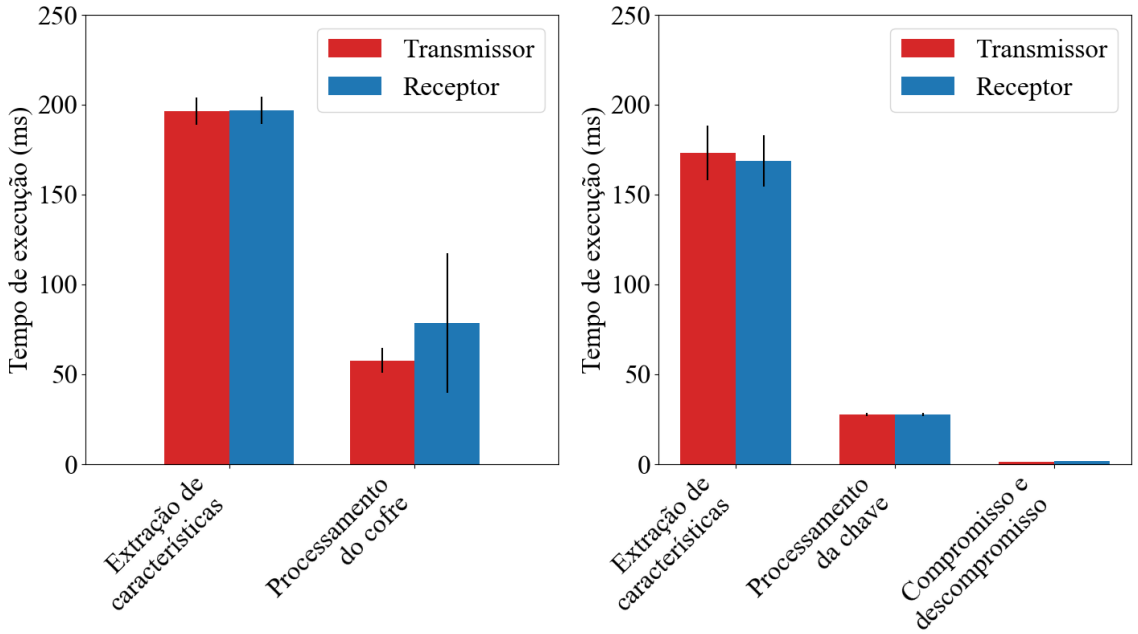
Concretizada a implementação dos protocolos escolhidos, foram aplicados os experimentos conforme a metodologia apresentada na Seção 4. Deste modo, esta seção aponta e discute os resultados obtidos. Foram realizadas comparações entre os protocolos EKA [Venkatasubramanian et al. 2008] e PSKA [Venkatasubramanian et al. 2009] tendo como essência da análise o consumo de memória, o tempo de execução, tempo de transmissão para o tamanho do pacote e os níveis de falsa aceitação e falsa rejeição.

Os gráficos ilustrados nas Figuras 3 e 4, exibem informações sobre a média ($\bar{x}$) e desvio padrão (σ) do tempo de execução em milissegundos (ms) dos protocolos para cada dispositivo (Transmissor e Receptor). São destacadas fases de cada um dos objetos de estudo pertinentes para a análise.

Na Figura 3a, são representados dois momentos. O primeiro é a extração da característica, o qual envolve todo o processo desde a coleta da amostra até a geração do vetor de características. Foram alcançados a média de $\bar{x}= 196.51ms$ e desvio de $\sigma = 7.6ms$ no transmissor. No receptor a média foi de $\bar{x} = 196.95ms$ e desvio de $\sigma = 7.62ms$. O outro momento é o processamento do cofre. Este, compreende a geração do polinômio e o trancamento do cofre no transmissor, com a média de $\bar{x} = 57.75ms$ e desvio de $\sigma = 6.9ms$. No receptor, a fase indica o destrancamento do cofre, alcançando os resultados de média $\bar{x} = 78.54ms$ e desvio de $\sigma = 38.93ms$. Analisando a Figura 3a, nota-se tanto no transmissor quanto no receptor que o tempo de extrair as características são semelhantes uma vez que processam o mesmo conjunto operações em ambos os dispositivos. Entretanto, ao observar o fase de processamento do cofre, destaca-se que o receptor possui um tempo médio de execução de 36% maior em relação ao transmissor. Tal sobrecarga de tempo ocorre pois, enquanto o custo de processamento pelo dispositivo transmissor compreende a geração do cofre, no dispositivo receptor engloba o destrancamento do cofre. Isso promove um custo computacional elevado, uma vez que envolve a filtragem dos pontos verdadeiros do cofre e a interpolação para reconstrução do polinômio. Computacionalmente abordando, a interpolação de um polinômio possui maior custo que sua produção. Além disso, o alto desvio padrão no receptor é um reflexo de que o tempo médio varia de acordo com a quantidade pontos utilizados para construir o cofre. Em cofres com poucos pontos, a interpolação é executada rapidamente. A quantidade de pontos varia de acordo com a quantidade de características extraídas, quanto mais características, mais pontos.

O tempo médio de execução das fases de extração de características, compromisso, descompromisso, e processamento da chave do protocolo EKA são exibidos na Figura 3b. A extração de característica compreende da coleta dos dados até geração do vetor de características. Os valores alcançados são: média $\bar{x} = 173.01ms$ e desvio de $\sigma = 15.17ms$ no transmissor, média $\bar{x} = 168.64ms$ e desvio de $\sigma = 14.27ms$ no receptor. A fase de compromisso e descompromisso engloba a geração das matrizes de *hash*,

Figura 3. Tempo médio de execução



(a) Fases - PSKA

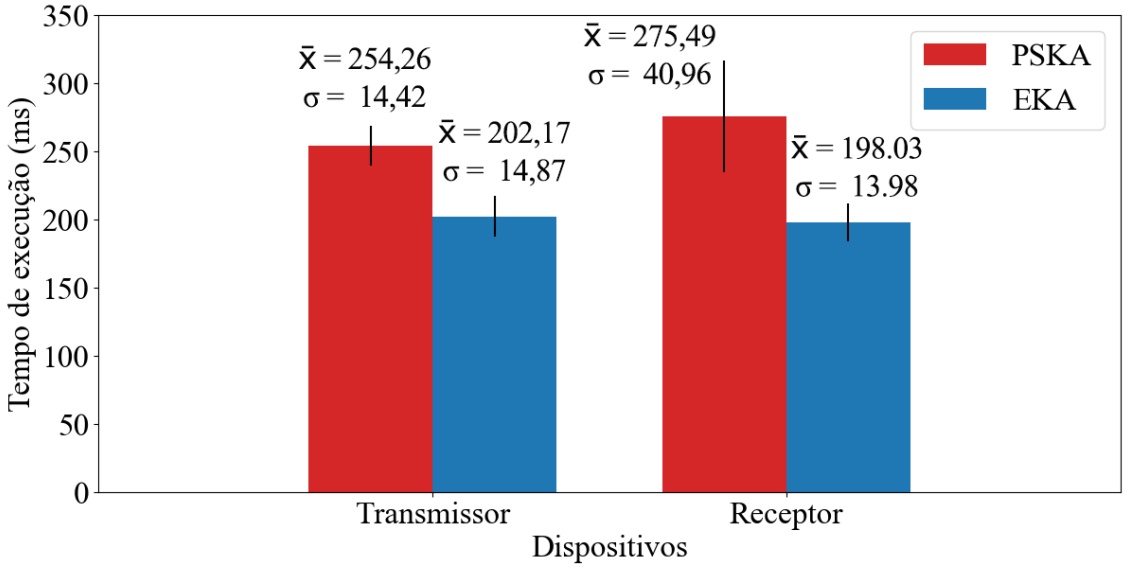
(b) Fases - EKA

média $\bar{x} = 1.43ms$ e desvio de $\sigma = 0.03ms$ no transmissor, média $\bar{x} = 1.76ms$ e desvio de $\sigma = 0.03ms$ no receptor. Já a fase de processamento da chave realiza o calculo da matriz de distância e extrai a chave, média $\bar{x} = 27.73ms$ e desvio de $\sigma = 0.78ms$ no transmissor, média $\bar{x} = 27.63ms$ e desvio de $\sigma = 0.78ms$ no receptor.

O custo total de ambos protocolos são apresentados na Figura 4, a qual exibe o tempo médio total de execução dos protocolos. Observa-se que o protocolo PSKA tem um tempo maior de execução, cerca de 25.8% a mais no transmissor e 39.1% no receptor. Isto ocorre devido às operações realizadas para construir o vetor de características e o destrancamento do cofre. Para construir o vetor no PSKA utiliza-se a FFT, com complexidade de tempo conhecida por $O(N\ log\ N)$, em conjunto com um algoritmo de detecção de pico (*local maxima*), com complexidade conhecida por $O(N)$. Como a geração do vetor de características no EKA envolve apenas a FFT, o tempo de criação é menor. Ademais, para realizar o destrancamento do cofre no PSKA, é utilizada uma operação de interseção ($O(N)$) e em seguida a interpolação de Lagrange ($O(N\ log\ N)$). Enquanto o processamento da chave no EKA gera uma matriz de distância de *hamming* ($O(N)$) e efetua uma busca por distâncias iguais a 0. Teoricamente a busca em uma matriz é mais custoso ($O(N^2)$), mas N nesse caso é conhecido como o número de blocos (Tem o valor fixado em 20). Então na prática apresenta um tempo menor pois o valor de N no PSKA depende da quantidade de pontos no cofre (durante os testes, a quantidade de pontos ultrapassou 200 na maioria das vezes).

O consumo de memória dos protocolos são representados nas Figuras 5 e 6. Assim como na avaliação do tempo médio de execução, a observação dos protocolos foram divididos em fases as quais já foram descritas anteriormente. O protocolo PSKA consumiu em média $\bar{x} = 135.61KB$ de memória com desvio de $\sigma = 0.7KB$ na fase de

Figura 4. Tempo médio de execução - PSKA x EKA



extração de características em ambos os dispositivos. Na fase de processamento do cofre, a média foi $\bar{x} = 16.45KB$ com desvio de $\sigma = 1.71KB$ para o dispositivo transmissor e $\bar{x} = 12.35KB$ com desvio de $\sigma = 1.28KB$ para o dispositivo receptor. Os resultados obtidos com o EKA foram idênticos em ambos dispositivos, sendo a média $\bar{x} = 144.0KB$ com desvio de $\sigma = 0.0KB$ na fase de extração de características, $\bar{x} = 10.52KB$ com desvio de $\sigma = 0.0KB$ na fase de processamento da chave e $\bar{x} = 6.11KB$ com desvio de $\sigma = 0.0KB$ na fase compromisso e descompromisso.

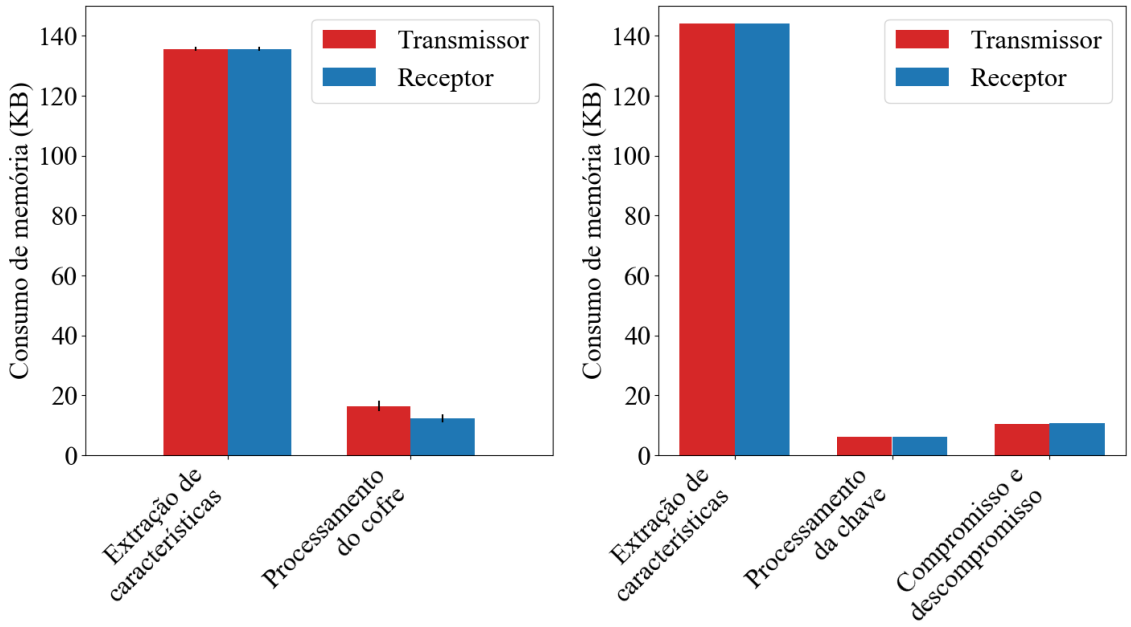
É notável que o consumo médio de memória são semelhantes quando são observados entre dispositivos em um mesmo protocolo, com exceção do transmissor no processamento do cofre ao avaliar o protocolo PSKA. O transmissor consome cerca de 33.2% mais memória que o receptor e isso se deve ao fato de ter que armazenar o polinômio gerado antes de construir o cofre definitivamente.

Observando o gráfico da Figura 6, é possível perceber que o consumo de memória do EKA é superior ao do PSKA, 6% a mais no transmissor e 8.8% a mais no receptor. Isso acontece devido ao EKA armazenar matrizes com os *hashes* de cada um dos blocos de características produzidas, ou seja, há duas matrizes armazenadas em cada um dos dispositivos. Uma matriz representando os blocos gerados no transmissor e uma matriz dos blocos gerados no receptor. Essas matrizes devem ser armazenadas para gerar a matriz de distância que posteriormente auxiliará na constituição da chave de autenticação. Enquanto isso, no PSKA são armazenados apenas o cofre com pontos falsos e verdadeiro que serão utilizados para dar origem à chave.

Desta forma podemos notar um *trade-off* entre tempo de execução e consumo de memória entre os protocolos. O PSKA lida com operações mais custosas computacionalmente em questão de complexidade de tempo com o processamento das características e processamento do cofre. O EKA processa menos as características, mas consume mais espaço com a utilização de matrizes.

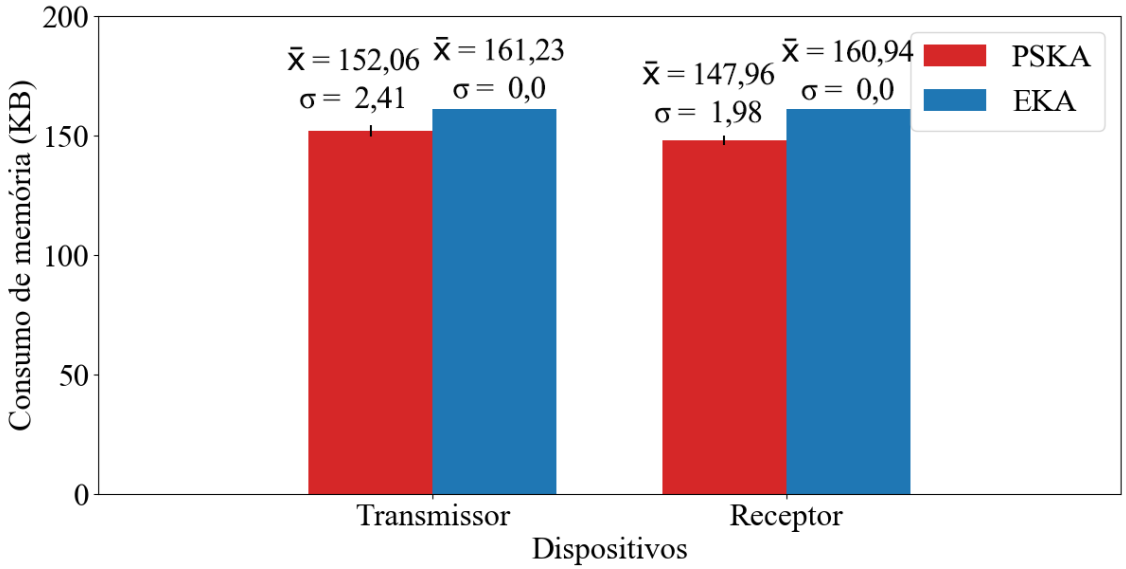
A análise de validação das implementações incluindo a taxa de falsa aceitação

Figura 5. Consumo médio de memória



(a) Fases - PSKA **(b) Fases - EKA**

Figura 6. Consumo médio de memória - PSKA x EKA



(FAR) e a taxa de falsa rejeição (FRR) pode ser contemplada na Tabela 1. A taxa de aceitação (AR) indica a porcentagem dos acordos que foram aceitos, ao somar AR com FRR, obtém-se o percentual que deveria ter sido aceito. Isso também vale para a taxa de rejeição (RR), RR somado a FAR indica a quantidade de acordos que deveriam ser rejeitados. Observa-se que o protocolo PSKA dispõe de uma taxa de falsa rejeição de 6,92% e falsa aceitação em torno de 11,3%. A taxa de falsa rejeição cresce de acordo com o tamanho do polinômio e em contrapartida a taxa de falsa aceitação diminui quando o tamanho do polinômio é maior. Além disso, o PSKA, dispõe da possibilidade de aceitar

uma requisição mesmo que os dispositivos não estejam completamente sincronizados. Outro fator que pode ser apontado é que a quantidade de características extraídas interfere nas taxas de FRR e FAR. Uma vez que são extraídas poucas características, haverá menos pontos no cofre, abrindo margem para descobrir mais facilmente os pontos verdeiros e consequentemente o polinômio que guarda o segredo. Se repararmos o EKA na Tabela 1, percebe-se que o mesmo atinge taxas incomuns de 0% de falsa aceitação e falsa rejeição. Isto ocorre baseado na igualdade dos dados no receptor e o transmissor. Desta maneira a mesma matriz de distância é gerada em ambos dispositivos.

Tabela 1. Análise do vetor de características - FAR e FRR

	AR	FRR	RR	FAR
PSKA	93.08%	6.92%	88.70%	11.30%
EKA	100%	0%	100%	0%

O *goodput* alcançado com as mensagens do PSKA foi de 1.63 acordos por segundo, considerando o tamanho da mensagem como 6656 bits e da mensagem de confirmação como 160 bits, totalizando 6816 bits de informação. Para calcular o *goodput* do EKA, foi considerado o tamanho da mensagem gerada em um dispositivo já que ambos os dispositivos fazem a troca simultânea das mesmas mensagens. O tamanho da mensagem de compromisso foi de 5344 bits e da mensagem de descompromisso foi de 288 bits, totalizando 5632 bits. Desta forma o *goodput* alcançado foi de 1.97 acordos por segundo.

O tamanho da mensagem no PSKA varia de acordo com a quantidade de características extraídas pois isso tem impacto direto no tamanho do cofre. Então foi considerado a média de 200 pontos (média obtida durante os testes) no cofre para realizar o cálculo. O EKA tem tamanho fixo de mensagem, porém elas devem ser transmitidas duas vezes, uma vez pelo transmissor e outra pelo receptor.

O fato da matriz gerada em ambos dispositivos serem iguais no EKA pode ser um problema. Por causa desse aspecto, é necessária uma sincronização mais apurada dos transmissores para implantação do protocolo, o que não acontece com o protocolo PSKA. É importante apontar que o PSKA não atinge o 0% de falsa rejeição ou falsa aceitação com a configuração que foi imposta, mesmo que os dados sejam iguais. Isso se deve ao fato de que as constantes que compõe o polinômio são geradas de forma aleatória e desta forma, ao realizar a interpolação dos pontos verdadeiros do cofre, o erro produzido pelo cálculo pode conduzir a uma falsa rejeição ou falsa aceitação.

Em resumo, o protocolo PSKA apresentou um tempo de execução mais elevado e em contrapartida consumiu menos memória que o EKA. Além disso, a análise de *goodput* indica que o PSKA realiza menos acordos por segundo devido ao tamanho da mensagem a ser transmitida. Apesar dessas diferenças, a análise de FAR e FRR demonstrou que o EKA é inflexível com relação a sincronização dos sensores. Portanto, considerando um cenário controlado, o EKA se sai melhor. Entretanto, em um ambiente real, o PSKA é mais adequado devido à sua maior predisposição à sincronização.

6. Conclusão

Neste trabalho foi apresentado uma avaliação empírica de protocolos de autenticação de usuários. A análise compreendeu tempo de execução, consumo de memória e

goodput. A avaliação foi realizada para as principais etapas do processo de autenticação. Destaca-se as etapas de extração de características e o processamento da chave ou cofre. Os resultados destacam que a etapa de extração de características é o motivo principal para o elevado consumo de memória e tempo em relação às demais etapas para ambos os protocolos. O PSKA apresentou uma média de tempo de execução maior do que o EKA, em contrapartida o EKA demonstrou um consumo de memória mais elevado, demonstrando um *trade-off* de desempenho e memória entre as estratégias utilizadas. Além disso, o PSKA apresentou um *goodput* menos satisfatório em relação ao EKA devido ao tamanho do cofre gerado. O tamanho do cofre nesse caso tem impacto direto no tamanho da mensagem de acordo de chaves.

Uma avaliação foi realizada envolvendo as taxas de falsa rejeição e falsa aceitação afim de validar as implementações dos protocolos. Apesar de ser apresentado que o EKA possui taxas de falsa rejeição e falsa aceitação melhores, essas taxas só são alcançadas quando os sensores estão sincronizados na coleta dos dados, o que não acontece em um cenário real. Nesse caso, o PSKA é mais aplicável em cenário real pois não exige uma sincronização com alto nível de apuração.

Considerando que a etapa de extração de característica é a mais crítica em ambos protocolos, há a expectativa que se desenvolvam trabalhos futuros pensando em soluções que melhorem essa etapa. Além disso, uma melhoria que pode ser realizada no EKA é a falta de predisposição para lidar com dispositivos sincronizados. Isso pode ser alcançado trabalhando com o vetor de características a ser gerado. E no PSKA poderia haver um limite de características extraídas, de modo que o tamanho do cofre seja menor e possibilitando um goodput melhor e um tempo de execução reduzido.

Referências

Ali, A. and Khan, F. A. (2015). Key agreement schemes in wireless body area networks: Taxonomy and state-of-the-art. *Journal of medical systems*, 39(10):115.

Cherukuri, S., Venkatasubramanian, K. K., and Gupta, S. K. S. (2003). Biosec: a biometric based approach for securing communication in wireless networks of biosensors implanted in the human body. In *2003 International Conference on Parallel Processing Workshops, 2003. Proceedings.*, pages 432–439.

Juels, A. and Sudan, M. (2002). A fuzzy vault scheme. In *Proceedings IEEE International Symposium on Information Theory,*, pages 408–.

Juels, A. and Wattenberg, M. (1999). A fuzzy commitment scheme. In *Proceedings of the 6th ACM conference on Computer and communications security*, pages 28–36.

Kalyakulina, A. I., Yusipov, I. I., Moskalenko, V. A., Nikolskiy, A. V., Kozlov, A. A., Kosonogov, K. A., Zolotykh, N. Y., and Ivanchenko, M. V. (2018). Lu electrocardiography database: a new open-access validation tool for delineation algorithms. *arXiv preprint arXiv:1809.03393*.

Khan, J. Y. and Yuce, M. R. (2010). Wireless body area network (wban) for medical applications. *New developments in biomedical engineering*, 31:591–627.

Maisel, W. H. (2010). Improving the security and privacy of implantable medical devices. *The New England journal of medicine*, 362(13):1164.

Marin, E., Argones-Rúa, E., Singelée, D., and Preneel, B. (2016). A survey on physiological-signal-based security for medical devices. *IACR Cryptol. ePrint Arch.*, 2016:867.

Nguyen, K. T., Laurent, M., and Oualha, N. (2015). Survey on secure communication protocols for the internet of things. *Ad Hoc Networks*, 32:17–31.

Poon, C. C. Y., Yuan-Ting Zhang, and Shu-Di Bao (2006). A novel biometrics method to secure wireless body area sensor networks for telemedicine and m-health. *IEEE Communications Magazine*, 44(4):73–81.

Rawat, P., Singh, K. D., Chaouchi, H., and Bonnin, J. M. (2014). Wireless sensor networks: a survey on recent developments and potential synergies. *The Journal of supercomputing*, 68(1):1–48.

Vale-Cardoso, A., Moreira, M., Coelho, K. K., Vieira, A., Santos, A., Nogueira, M., and Nacif, J. A. M. (2020). A low-cost electronic system for human-body communication. *Electronics*, 9(11):1928.

Venkatasubramanian, K. K., Banerjee, A., and Gupta, S. K. (2008). Ekg-based key agreement in body sensor networks. In *IEEE INFOCOM Workshops 2008*, pages 1–6. IEEE.

Venkatasubramanian, K. K., Banerjee, A., and Gupta, S. K. S. (2009). Pska: Usable and secure key agreement scheme for body area networks. *IEEE Transactions on Information Technology in Biomedicine*, 14(1):60–68.

Wang, H., Fang, H., Xing, L., and Chen, M. (2011). An integrated biometric-based security framework using wavelet-domain hmm in wireless body area networks (wban). In *2011 IEEE international conference on communications (ICC)*, pages 1–5. IEEE.

Wang, W., Hua, K., Hempel, M., Peng, D., Sharif, H., and Chen, H.-H. (2010). A stochastic biometric authentication scheme using uniformed gmm in wireless body area sensor networks. In *21st Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1620–1624. IEEE.