Abordagens acerca da segurança cibernética em aplicação de ponto eletrônico em dispositivos móveis

Filipe da Silva Sousa
Esp. Segurança Cibernética
FACENS
Sorocaba, Brasil
filipe.sousa@facens.br

Gelson José de Almeida Filho
Esp. Segurança Cibernética
FACENS
Sorocaba, Brasil
gelson.filho@facens.br

Rodrigo Augusto Almeida de Camargo
Esp. Segurança Cibernética
FACENS
Sorocaba, Brasil
rodrigo.camargo@facens.br

Otávio Marelli Marques
Esp. Segurança Cibernética
FACENS
Sorocaba, Brasil
otavio.marques@facens.br

Abstract*—The objective of this study is to develop security features for a software time clock system, having as a starting and fundamental point the mapping of the valuable/sensitive data along with the appropriate features in relation to access, storage, reproduction, transmission and data elimination. To mitigate vulnerabilities and propose a cybersecurity approach to the mapped data, the present papper also brings the study and development of security modules, where the first is the use of machine learning, through the K-NN - K nearest neighbors algorithm, performing the detection of malware in equipment with Android system and increasing their security level against possible threats, using at first a public database for training which is contextualized during the papper for better understanding of their data sets. The second security module studied was the use of encryption and data signing for communication between the application and the central server using the Diffe-Hellman protocol to ensure the confidentiality and authenticity of the information, aiming the use of algorithms suitable for devices with limited resources. The last security module that is presented is the use of a hash system to point out modifications in the database, with or without justifications, in order to assist in the identification of fraud. At first, a performance comparison is made between hash generation algorithms, such as MD5, SHA1, SHA224, SHA256, SHA384, and SHA512. After all the development finally the results are presented and discussed considering also their limitations.

Palavras Chaves*—controle de ponto, requisitos de segurança, mapeamento de valores, vulnerabilidades, Machine Learning, K-NN, Malware, Android, Diffie-Hellman, criptografia, X25519, Ed25519, AES-GCM, Hash, MD5, SHA256

I. INTRODUÇÃO

Até poucos anos atrás acompanhar a jornada e trabalho de funcionários externos era uma tarefa complexa. Porém, com o desenvolvimento de aplicações de controle de ponto isto acabou se tornando um pouco mais simples. Em sua maioria, os pontos eletrônicos podem ser instalados no celular do próprio funcionário permitindo acesso à serviços e documentos [1].

O controle de ponto por aplicativo geralmente funciona por meio de um sistema de ponto online, desenvolvido para captar as marcações e informações do registro de ponto dos funcionários [1]. O aplicativo é apenas uma parte deste grande software de ponto, que pode ser dividido em três partes: Registro de ponto, Tratamento de ponto e Gestão de ponto [1].

Além disso, a legislação trabalhista contempla essa modalidade, desburocratizando a rotina das empresas [2]. Aplicativos de controle de ponto foram autorizados pela Portaria 373 do MTE de 2011, que regulariza o uso de sistemas alternativos de controle de jornada. Essa lei determina os requisitos obrigatórios para o app de controle de ponto, como a impossibilidade de marcação automática e a emissão de comprovantes de registro para o empregado, entre muitos outros [2].

Conduzindo a situação para a perspectiva da segurança cibernética, todo o processo deve ser visto com foco em proteger o que há de valor, dessa forma, é necessário se conhecer a importância das diversas informações recebidas, utilizadas, armazenadas e transmitidas pela instituição [9]. É a isto que parte deste trabalho se propõe, uma metodologia para classificação das informações de uma instituição. Isto visa facilitar o trabalho de segurança, permitindo se definir o nível de segurança que deve ser utilizado no armazenamento e transmissão das informações por esta metodologia categorizadas.

Uma vez que as informações que trafegam pelo processo de ponto eletrônico são classificadas com o mais alto nível de proteção, devem-se considerar técnicas para garantir confidencialidade e integridade destas informações, tais como, medidas de segurança para o aplicativo, reforço de segurança no processo de transferência de dados e medidas para validação do recebimento destes dados.

Outrossim, uma abordagem que bem possibilita maior segurança cibernética para aplicativos de ponto eletrônico é a da detecção de *malwares* em dispositivos com Android, visto que grande parte dos dispositivos móveis ainda compartilham de tal sistema [17]. Considerando isso, esse artigo propõe também a detecção de *malware* em Android utilizando um algoritmo de *Machine Learning* voltado à classificação, conhecido como *K-NN – K Nearest Neighbors*. Analisar a possível presença de *malwares* nos dispositivos que estão hospedando a aplicação é somar à segurança atrelada à mesma, por isso foi escolhido o banco de dados do projeto "*DroidFusion*" de Yerima e Sezer [18] para construir o conhecimento sobre como trazer à prática a abordagem da aplicação do algoritmo inteligente para detecção de *malware* em dispositivo móvel.

Para garantir segurança em canais de comunicações é possível adotar metodologias de criptografia que agem na garantia da confidencialidade e sistemas de assinatura digital, como algoritmos RSA, DSA, ECDSA, os quais se baseiam em problemas matemáticos tal como o problema de logaritmo discreto [36], a fim de garantir a autenticidade. Visando alcançar a meta de segurança no canal de comunicação foi estudado métodos já aplicados em soluções de referência no mercado, como a criptografia fim-a-fim do *WhatsApp* também conhecida como *Signal Protocol* [32], a fim de desenvolver uma base de conceito para o módulo de segurança.

Um dos principais pilares da segurança da informação é a integridade, sendo está definida pela ISO/IEC 17799:2005 como salvaguarda de exatidão e completeza da informação e métodos de processamento [39], sendo, portanto, necessária de se levar em consideração diante desta abordagem acerca de melhorias na segurança cibernética de um aplicativo de ponto eletrônico. Pensando-se nesta necessidade, optou-se pela utilização de funções *hash* a fim de que se possa validar a integridade dos dados, visto que esta é uma das funções exercidas por esta tecnologia [40], posto que estes poderão ser manipulados de forma indevida no processo de validação e conferência das informações.

O artigo prosseguirá na seguinte sequência: Na seção II, abordaremos o referencial teórico particularmente para cada abordagem neste trabalho representada. Na seção III os detalhes de cada abordagem e metodologias aplicadas serão explanados. Na seção IV reuniremos os resultados, discussões e limitações que podem gerar ideias para trabalhos futuros. Por fim concluiremos, na seção V, todas abordagens visando integrá-las ao bem comum pertinente à segurança cibernética em aplicações de ponto eletrônico em dispositivos móveis.

II. REFERENCIAL TEÓRICO

Dada as quatro abordagens que este trabalho visa dissertar, vários são os trabalhos que foram necessários a leitura. Tais leituras e embasamento teórico precisaram de efetivação via testes práticos. Dada a importância das referências teóricas à nível de conhecimento, esclarece-se a seguir, para cada abordagem, as elucidações teóricas estabelecidas neste trabalho.

A. Mapeamento de dados de maior valor

Independentemente de setores ou de mercados, os dados se tornaram a força motriz que promove o desenvolvimento de uma variedade de estratégias nas empresas [3].

Os dados, na prática, são brutos, desorganizados, não analisados, ininterruptos e não relacionados, usados em diferentes contextos. Os dados, em essência, carecem de interpretação para que se tornem, finalmente, uma informação. Sem serem interpretados ou tratados, os dados tornam-se sem sentido, a menos que recebam um propósito ou uma orientação para adquirir significado [3].

Quando esses dados são analisados, estruturados e recebem compostura ou contexto para torná-los úteis, encontramos "informações". A informação remonta ao ato de informar, usada principalmente no contexto de conhecimento, instrução e educação. Em essência, a informação é sistemática, filtrada e útil [3]. A informação é um ativo que, como qualquer outro ativo importante para os negócios, tem um valor para a organização e consequentemente necessita ser adequadamente protegida [4].

Ao pesquisar os aplicativos de controle de ponto é comum notar que, em sua maioria são ferramentas para aquisição de dados de um colaborador e geração de relatórios e arquivos. Alguns aplicativos ainda disponibilizam ao colaborador acesso à documentos referentes a encargos e pagamentos.

Reunir e administrar os documentos do departamento pessoal, sendo estes, toda a documentação dos funcionários da empresa, compete ao setor de Recursos Humanos. A partir da entrada em vigor das mudanças na legislação trabalhista, a área ganhou ainda mais relevância, uma vez que a reforma deu liberdade para empregador e empregado acordarem diversas condições de serviço. Dessa maneira, fiscalizar e manter a ordem da documentação se tornou uma parte ainda mais importante da organização de uma empresa, para não gerar passivos trabalhistas [5].

A organização e o correto armazenamento destas informações são determinantes para dar celeridade e confiabilidade aos processos da empresa [5].

A preocupação com a segurança nos leva ao trabalho de assegurar que a informação seja classificada em termo do seu valor, requisitos legais, sensibilidade e criticidade para evitar modificação ou divulgação não autorizada [6].

As principais normas que embasam esta temática, e utilizadas para compor o modelo proposto são 16167:2020 Diretrizes para Classificação da Informação, 27002:2013 Código de prática para controles de Segurança da Informação. Buscam-se também os repositórios citados na bibliografia com a finalidade de consolidar uma efetiva aplicabilidade.

B. Detecção de malware com machine learning

Um *malware* pode ser definido como um programa inserido de maneira oculta em um sistema computacional com foco em degradação e comprometimento da integridade, confiabilidade e disponibilidade dos dados de vítimas, aplicações ou sistemas [10].

Várias são as classificações possíveis para um *malware*. De acordo com o CERT.br [11], vírus, *worms*, *trojans*, *botnets*, *spywares*, *ransomwares e rootkits* são alguns dos principais tipos de códigos maliciosos presentes em sistemas computacionais. Vírus se propagam, após uma execução humana, tornando-se parte de outros arquivos e criando

copias de si mesmo [12]. Já os *worms* não precisam de tal interação humana para propagar sua autoinfecção [13]. Os *trojans*, também conhecidos como cavalos de Troia, podem executar ações maliciosas com foco em permitir ao atacante um acesso privilegiado ao sistema bem como acessar informações confidenciais [13]. Não menos comum, os *botnets* são mecanismos de invasão que formam redes de computadores "zumbis", que possibilitam ao invasor o acesso remoto [14]. *Spywares* são aplicações maliciosas voltadas à captura de informações de teclado (*keylogger*), posição do cursor e tela do monitor (*screenlogger*) ou ao oferecimento indevido de anúncios e propagandas sem consentimento do usuário (*adwares*) [15]. Muito comum nos dias atuais, os *ransomwares* envolvem geralmente a criptografia dos arquivos do usuário a fim de gerar enorme perda de dados e consequentemente prejuízos de diversas naturezas [16]. Ademais, o conjunto de softwares exploradores que visam esconder atividades e comportamentos de um sistema é conhecido como *rootkit*, onde a junção da palavra "*root*" (privilégios administrativos) e "*kit*" (conjunto de programas) bem esclarecem o funcionamento deste malware [12].

De maneira geral, não é errado dizer que o Android é apenas uma máquina virtual Java rodando sobre o kernel do Linux, viabilizando o desenvolvimento de aplicações Java a partir de várias bibliotecas e serviços [17]. Considerando esta forte relação entre o Android e o Linux é justificável dizer que arquivos executáveis do Android possuem assinatura (cabeçalho) semelhante a arquivos de extensão ELF – Executable e Linkable Format, portanto faz-se viável a obtenção de dados de dispositivos Android considerando os arquivos com tal assinatura, além de informações relacionadas à chamada de API – Application Programming Interface e ao arquivo de manifesto de permissões do Android [18].

Baseando-se nesses dados importantes do Android, o projeto "DroidFusion" de Yerima e Sezer [18] bem os utilizou e os disponibilizou publicamente. A amostra de dados utilizada pelos autores nada mais é do que a mescla entre outras amostras públicas amplamente utilizadas: a do projeto Android Malgenome [19], a DREBIN [20] e outra coleção de amostras fornecidas pela Intel Security (anteriormente conhecida como McAfee). Os autores, Yerima e Sezer [18] ainda propõem um analisador estático automatizado para extração de recursos desenvolvido em Python. Resumidamente eles extraem permissões e intenções do arquivo de manifesto de aplicativos após descompactação com a biblioteca AXMLprinter2. Já as chamadas de API são extraídas utilizando engenharia reversa dos arquivos .dex por meio do desmontador Baksmali. O analisador automático de Yerima e Sezer [18] verifica a presença de arquivos .dex, .jar, .so e .exe potencialmente maliciosos presentes nos aplicativos bem como comandos Linux igualmente maliciosos. Tais atributos coletados possuem considerável valor de discriminação sobre ser ou não um malware ajudando em muito a aplicação em algoritmos inteligentes como bem comprova trabalhos anteriores [21].

Ademais, bibliotecas de terceiros, menos importante, são excluídas no analisador automático, utilizando-se de uma lista de bibliotecas genéricas e populares obtidas em [22]. Yerima e Sezer [18] geraram um excelente conjunto de dados, já sumarizado e normalizado, portanto como o foco deste trabalho não é estender-se sobre os algoritmos de *machine learning*, mas sim abordar o uso da tecnologia com foco na segurança em dispositivos móveis com aplicativo de ponto eletrônico, optou-se por não desenvolver outro conjunto de dados, mas sim utilizar o já consolidado conjunto de dados dos autores citados.

Se por um lado Yerima e Sezer [18] utilizaram quatro diferentes classificadores para comparar o desempenho de cada um, aqui utilizamos o K-NN – K Nearest Neighbors para abordar o quão eficiente um algoritmo classificador genérico pode ser em detectar malware em um dispositivo Android. A escolha pelo K-NN se dá pela já importância dada em outros trabalhos relacionado à detecção de malware, como no trabalho de Stiawan, Daely, Heryanto, Afifah, Idris e Budiarto [23].

C. Criptografia para segurança na comunicação

Dentre as vulnerabilidades que o sistema alvo de estudo pode possuir, a troca de dados entre o Aplicativo no celular do usuário e o servidor de processamento é considerado um ponto fraco, uma vez que se considera que tais informações estarão trafegando na internet, um meio não seguro e susceptível a ataques como do Homem no Meio (do inglês *Man in the Middle* - MITM).

Onde ataques desse gênero se caracterizam pela interceptação dos dados enviados pelo usuário para o servidor, de modo que possam ser registrados e alterados sem o consentimento das partes. Esse tipo de ataque pode ocorrer em diversos cenários, onde o atacante atuará por meio de agentes intermediários como: roteadores, servidores de proxy ou DNS [27].

A fim de trazer maior segurança à comunicação Servidor–Aplicativo, é proposta a implementação de um sistema de criptografia por chave simétrica, porém com uso do protocolo *Diffie-Hellman* para geração dessa chave simétrica a partir de chaves públicas fornecidas pelas pontas.

O algoritmo DH, como também é conhecido o protocolo *Diffie-Hellman*, combina o valor secreto (chave privada) com um valor público (chave pública) para gerar a chave combinada. Essa troca de segredo de maneira segura através da chave pública basicamente só é possível graças a operação de módulo realizada durante a geração do valor público [28].

Para aplicação do algoritmo DH, optou-se pelo uso do *Diffie-Hellman* com curvas elípticas (ECDH), em específico a função X25519, apresentada em 2006 por [29]. Em [29] leitor é introduzido sobre a Curva25519, sua aplicação e discussão sobre a eficiência dela.

Uma das características de eficiência em destaque é a alta performance do modelo proposto por [29]. Uma vez que tal solução será destinada a uso em aplicativos de celular que se caracterizam pelo uso de baterias e possuem limitações de hardware e processamento por conta de consumo de energia.

Além da eficiência, outra característica importante descrita por [29] é a não variação de tempo de execução sem a implementação de recursos de segurança ao algoritmo, de modo a não comprometer seu desempenho [29].

Tal característica torna a função resistente a Ataques de Tempo, uma vez que esses ataques consistem em observar o

tempo que um sistema leva para responder a requisições feitas pelo atacante, seja localmente ou via internet [30 - 31].

Em complemento na garantia da confidencialidade, o projeto visa aplicar medidas que trabalhem no quesito de autenticidade dos dados enviados, sendo estudado a aplicação de funções do tipo Ed25519. Essa função trabalha com a mesma curva 25519, gerando chaves com bom nível de segurança e baixa demanda de processamento, uma vez que ela foi elaborada visando a implementação em equipamentos com recursos limitados [37].

Os algoritmos de assinatura por curva elíptica seguindo o modelo *Edwards-curve Digital Signature Algorithm* (EdDSA), apresentam duas variantes, os modelos denominados como *PureEdDSA* e *HashEdDSA*, onde o primeiro caso trata-se de um modelo de assinatura por passagem única e o segundo traz a característica de resistência a colisão de *hash*, assim como SHA-512 [38]. Tendo em vista a limitação processamento do dispositivo móvel a opção que se faz mais viável é a de *PureEdDSA*, por tratar a assinatura de entrada única, todavia sem garantia de resistência a colisão de *hash*.

D. Hashing de verificação

Uma função *hash* pode ser compreendida como um algoritmo capaz de realizar cálculos matemáticos que transformam dados de tamanhos variáveis em uma saída de tamanho fixo [40], sendo utilizada para converter desde *strings* de texto, a arquivos, em uma saída específica de acordo com cada tipo de função *hash* utilizada [40], sendo este algoritmo de direção única, não podendo ser revertido por meio de cálculos ou outros algoritmos, pois a intenção do uso desta tecnologia é fazer com que essa saída única possa ser utilizada para verificar a integridade dos dados, visto que qualquer alteração, por menor que seja, deve a alterar completamente a mensagem de saída [41].

De acordo com [41] dentre as principais aplicações das funções *hash*, encontram-se: assinar digitalmente uma mensagem, verificar a integridade dos arquivos, e realizar a verificação de senhas. Ao comparar o valor gerado pela função *hash* aplicada na senha digitada por um usuário com os *hashes* presentes em um banco de dados, pode-se se certificar de que o usuário inseriu a senha correta [40]. Para que estas características sejam efetivas, espera-se que as *hashes* sejam geradas de forma unidirecional, tornando a reversão dessa chave gerada uma tarefa praticamente impossível, resistentes à segunda pré-imagem, ou seja, não se podem gerar dois valores iguais de saídas diferentes, e que apresente resistência à colisão, de modo que diferentes entradas não acabem por gerar a mesma saída [41].

Dentre os mais conhecidos algoritmos de *hashing* existentes atualmente, encontram-se o algoritmo MD5, que recebe uma entrada de dados e retorna uma saída de 128 bits, com um tamanho de 32 dígitos, tendo sido desenvolvido em 1990 e não sendo recomendado, visto que este algoritmo já foi quebrado, e o SHA256, que recebe uma entrada de dados e retorna uma saída de 256 bits, com um tamanho de 64 dígitos [42]. Existem ainda diversos outros algoritmos, como os pertencentes à família SHA-1, SMASH, RIPMED, Whirlpool [43], porém neste trabalho será dada maior atenção aos dois descritos anteriormente.

III. SEGURANÇA CIBERNÉTICA EM PONTO ELETRÔNICO

Com o desígnio de realçar o fluxo do funcionamento de uma aplicação de ponto eletrônico foi desenvolvido, no *software* Microsoft Threat Modeling Tool, o *Threat Model* concernente ao caminho das informações desde o dispositivo final do colaborador até o servidor da empresa onde elas são finalmente processadas. O modelo elaborado, que pode ser visto na figura 1 no rodapé desta página do trabalho, auxilia o entendimento dos parágrafos a seguir no que tange o engajamento em mitigar as possíveis ameaças cibernéticas à aplicação.

Ademais, antes de discorrer acerca da metodologia seguida em cada um dos testes práticos realizados, é importante explanar que, para o desenvolvimento dos algoritmos e testes de validações, foi utilizado um notebook com processador Intel Core i7-10750H (2.6 GHz, 2592 MHz, 6 núcleos e 12 processadores lógicos), 16GB DDR4, SSD NVMe 475GB com o sistema operacional Windows 11 Pro da Microsoft. Todos os códigos foram desenvolvidos utilizando o Anaconda Navigator 2.1.4 e os códigos inteiramente desenvolvidos no Visual Studio Code versão 1.71.2 com auxílio da extensão do Jupyter Notebook versão 2022.8.1002431955. Particularmente para os testes relativos à "detecção de malware com *machine learning*', algumas bibliotecas foram utilizadas: Pandas, Numpy, matplotlib, seaborn, mlxtend e a mais importante (que oferece todo o suporte para algoritmos de aprendizado de máquina) que é a *Scikit-learn*. Já nos testes voltados à "criptografia para segurança na comunicação", as bibliotecas foram: *cryptography*, *typing*, *OS*, *shutil* e *binascii*, sendo a *cryptography* a de maior relevância por prover funções voltadas ao campo de criptografia. Por fim, nos testes voltados ao "*hashing* de verificação", as bibliotecas do Python utilizadas foram: Pandas, assim como nos testes de detecção de malware, Timeit, para verificação do tempo que cada função *hash* levou para ser executada no banco de dados e a Hashlib, responsável por realizar a aplicação das funções *hash* que foram testadas.

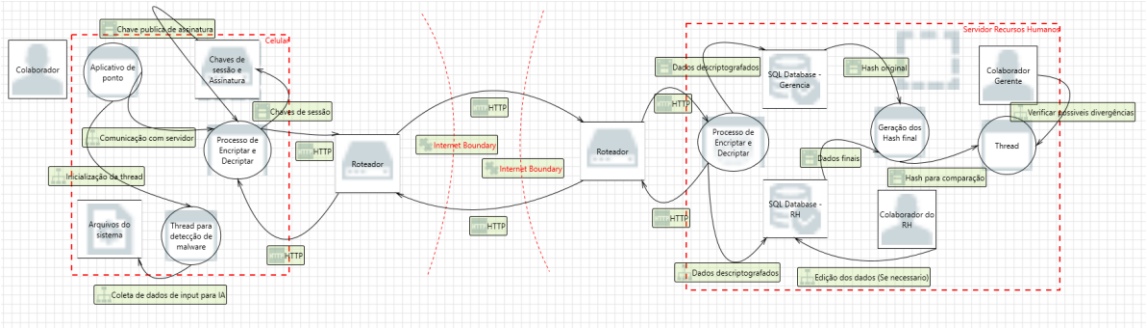

Fig. 1 *Threat Model* da aplicação do ponto eletrônico [Imagem do autor]

A. Mapeamento de dados de maior valor

Mendes [7] entende que a utilização massiva de dados pessoais por organismos estatais e privados, a partir de avançadas tecnologias da informação, apresenta novos desafios ao direito à privacidade. A norma visa assegurar que a informação receba um nível adequado de proteção, de acordo com a sua importância para a organização [6].

A Classificação fornece às pessoas que lidam com informações uma indicação concisa de como tratar e proteger a informação. A criação de grupos de informação com necessidades de proteção semelhantes e especificação dos procedimentos de segurança da informação que se aplicam a todas as informações de cada grupo, é um facilitador. Esta abordagem reduz a necessidade de avaliação de risco e a customização personalizada de controles caso a caso [6].

Santos e Rodrigues [8] redigem um documento de cunho corporativo com a finalidade de complementar as Políticas de Segurança da Informação de uma organização privada do mercado de manutenção e *facilities*, definindo as diretrizes para o devido armazenamento, manuseio e descarte de informações. Os autores descrevem inicialmente as responsabilidades de cada parte interessada presente na Política, sendo estes a organização, o encarregado de proteção de dados, colaboradores, clientes e usuário de dados. Estabelecendo diretrizes para armazenamento, anonimização e descarte.

Reis, Mota e Oliveira [9] especificam que os níveis de segurança devem ser descritos com base na análise dos três pilares da segurança da informação, e por sua vez apresenta a implementação da classificação, exemplificando o tipo do documento e seu respectivo procedimento. Os autores acreditam que com uma boa classificação das informações a organização não só poderá ter uma boa e otimizada política de segurança da informação, como também terá outros benefícios. Ela poderá conhecer melhor os seus processos, pois se verá forçada a fazer um inventário das informações.

Dessa forma, baseando em publicações e normas existentes até o devido momento, cria-se um modelo de segurança da informação para complementar a Política de Segurança da Informação, de simples entendimento e fácil aplicação, de forma a atingir pessoas, processos e tecnologias, a fim de definir as diretrizes para a devida classificação, rotulagem e manuseio das informações, de acordo com a sua sensibilidade e criticidade para a instituição, visando o estabelecimento de níveis adequados de proteção.

B. Detecção de malware com machine learning

Considerando o conjunto de dados já consolidado de Yerima e Sezer [18] é necessário à sua aplicação dentro de um algoritmo que execute a classificação pelo método K-NN com auxílio principalmente da biblioteca Scikit-learn. O K-NN costuma apresentar bons resultados em trabalhos relacionados à classificação assim como também seu uso abre amplas possibilidades de otimização e enfoques [24].

Após leitura do conjunto de dados e criação de um *dataframe* pela biblioteca pandas, há a necessidade de separar a coluna onde encontra-se a classificação supervisionada e as outras 215 colunas onde se encontram os atributos do conjunto de dados. No passo seguinte o modelo de seleção para treino é atribuído utilizando-se aleatoriamente de 20% de todo o conjunto de atributos e classificações supervisionadas para testes e os 80% restante dos dados para treinamento, essa escolha foi baseada no Princípio de Pareto que basicamente afirma que 80% dos problemas decorrem de 20% das causas [25]. A utilização dessa regra potencialmente alavanca o resultado de algoritmos inteligentes [26].

Então o treinamento do algoritmo K-NN pode ser finalmente executado permitindo a captura da acurácia, matriz de confusão e previsões de classificação. Também se torna viável, a partir de tal momento, a otimização dos parâmetros K e métrica utilizada na fase de treinamento. Nessa fase de treinamento, após vários testes empíricos, foi possível obter os melhores parâmetros baseado na acurácia que o algoritmo retornou, ao utilizar-se da métrica da Distância euclidiana com “K” vizinhos iguais a 4. Tais configurações são muito utilizadas e fielmente retornam bons resultados para conjunto de dados sumarizados e normalizados como os em questão [23].

A aplicação que seria muito útil, na integração desta abordagem com o projeto do ponto eletrônico, seria a de, ao final do treinamento, apenas entrar com amostras de conjunto de dados em que se deseja analisar a presença ou não de malware. Esta forma de validação dos dados também é amplamente utilizada em aplicações com o algoritmo K-NN [24].

C. Criptografia para segurança na comunicação

Tendo em vista de que a troca de informações anteriormente classificadas será realizada por um meio público, a internet, é necessário que seja implementada técnicas de segurança para garantir a confidencialidade dos dados. A princípio a criptografia se mostra como um caminho promissor, todavia se vê a necessidade do uso de técnicas seguras para a implementação dessa ferramenta, uma vez que a criptografia simétrica depende de uma chave que também será trocada por meios públicos, sendo exposta da mesma maneira que os dados.

Em decorrência dessa insegurança, se faz necessário o uso da técnica de *Diffie-Hellman*, que tem a proposta de proteger a chave secreta viabilizando a transferência em ambientes públicos.

Visando a implementação de um sistema que garanta a confidencialidade dos dados transitados no meio público (internet), foi proposto a elaboração de uma demonstração de um sistema de troca de chaves do tipo ECDH utilizando a função X25519 para chaves de sessão e a função Ed25519 para o par de chaves de assinatura.

Em uma primeira comunicação de um dispositivo com servidor raiz o processo deve ocorrer de maneira única, onde inicialmente é gerado o par de chaves da sessão atual, e no celular do colaborador o par de chaves de assinatura, através da função Ed25519, assumindo que o servidor principal já gerou suas chaves de assinatura anteriormente.

Uma vez que a sessão for concretizada com a troca das chaves públicas, o dispositivo móvel deve enviar sua chave pública de assinatura ao servidor e armazenar a chave de assinatura provida do servidor, para que haja a autenticação de remetentes. Tais chaves de assinatura geradas e recebidas são armazenadas na memória não volátil dos dispositivos de maneira independente da sessão, sendo que a chave privada de assinatura é criptografada antes de ser armazenada.

Como brevemente mencionado no caso da sessão de primeiro contato, em cada sessão deve ser gerado um par de

chaves de sessão, através da função X25519, em ambas as pontas (Servidor e Celular) a qual será válida apenas durante a conexão atual, sendo que ao termino dela, tal par deve ser eliminado para garantir a segurança dos dados trafegados e das futuras conexões, um conceito já explorado de diversas maneiras, tendo como exemplo o serviço de troca de mensagens WhatsApp, o qual foi submetido a uma análise de segurança no estudo de [32].

Analisando o caso do estabelecimento de sessão, uma vez que uma das partes envia sua chave pública de sessão e recebe a do destinatário, é iniciado o processo de criação da chave secreta, que será utilizada para encriptar e decriptar os dados a serem enviados. O processo se inicia com a geração da chave secreta compartilhada, onde é executado a função de geração da chave secreta compartilhada, a partir da chave pública recebida e privada local, e por fim é gerada uma chave final com tamanho fixo de 32bytes através da função HKDF, afim de gerar uma chave final forte para o uso em criptografia e para adotar ao apelo de padronização dos mecanismos KDF, como citado por [33].

Com a chave geral final gerada pela função HKDF é possível iniciar o processo de encriptação dos dados a serem enviados. Para tal processo optou-se pelo uso do algoritmo *Advanced Encryption Standard* (AES), um algoritmo de criptografia que trabalha processando blocos simétricos de 128 bits a serem cifrados com uso de chaves de 128, 192 e 256 bits [34], em conjunto com módulo *Galois Counter Mode* (GCM) que possui como maior diferencial a garantia da confidencialidade, cifrando as informações, e autenticidade, gerando uma *tag* de "assinatura" da mensagem ao termino do processo [35].

Uma vez que os dados são encriptados pela função AES-GCM, é gerado como saída: os próprios dados encriptados, o vetor de inicialização usado na função e a *tag* de assinatura do documento encriptado. Em seguida a mesma *tag* é assinada com a chave privada de assinatura presente no dispositivo para que então os dados sejam enviados ao destinatário final.

Observando pelo lado receptor, o mesmo deverá a princípio verificar a assinatura dos dados recebidos, inserindo na função de verificação, a assinatura, a chave pública de assinatura do remetente e o dado assinado, uma vez que passado pela função de verificação da assinatura os estarão aptos a serem decriptados para a análise final. A função que recupera os dados encriptados funciona de maneira similar à que encriptou, tendo como entrada, os dados cifrados, o vetor de inicialização e a *tag* de assinatura da função AES-GCM. Os dados só serão obtidos se a *tag* de assinatura corresponder aos dados cifrados, caso contrário a operação será interrompida e indicado que houve modificações nos dados, comprometendo a integridade.

Dessa forma o ciclo será finalizado no momento em que a sessão entre os dispositivos for encerrada, seja por opção do usuário ou por tempo de expiração, nesse momento, como já mencionado, as chaves de sessão são apagadas por definitivo, restando apenas as chaves destinadas a assinatura.

D. Hashing de verificação

Para realizar a aplicação da função *hash* nos dados disponibilizados pelo aplicativo de controle de ponto, foi utilizada a biblioteca hashlib, que permite a implementação de diferentes tipos de algoritmos de *hash*, como o SHA1, SHA224, SHA256, SHA384, SHA512 e outros mais antigos como o MD5, e a biblioteca timeit, que permite a medir o tempo de execução de um trecho de código, a fim de verificar a diferença de tempo entre as funções *hashes* aplicadas.

Para fins de testes, foram feitas comparações utilizando cada uma das variações de algoritmo presentes na biblioteca hashlib [406], avaliando-se o tempo de resposta e a viabilidade de aplicar estas funções, bem como possíveis critérios de escolha do algoritmo com base na aplicação que se deseja verificar, sendo neste caso, para fins de validação da integridade dos arquivos.

Inicialmente a função *hash* é aplicada no banco de dados do aplicativo, que é enviado utilizando-se a criptografia para a segurança na comunicação, e ao ser recebido pelo servidor cliente, cria-se então uma cópia sem a *hash* original, que é utilizada por usuários com acesso mais limitado do que o usuário que possui o arquivo original, com a *hash* gerada no aplicativo. Esta cópia pode ser alterada, caso haja necessidade de correção de algum registro no controle de ponto, e posteriormente é enviada ao usuário que possui acesso mais elevado. O algoritmo de *hashing* é então aplicado novamente nesta cópia, e se realiza uma comparação entre as *hashes* geradas originalmente, e as *hashes* geradas na cópia. Através desta comparação, pode-se identificar quais registros foram alterados, e então averiguar as justificativas inseridas no banco de dados para tais modificações. Sem que haja justificativas plausíveis, torna-se evidente que houve alterações não autorizadas, permitindo assim que haja um sistema de supervisão sobre essas informações.

Faz-se necessário que a função *hash* escolhida para ser aplicada a este projeto possa ser aplicada de forma rápida, visto que um algoritmo que demore muito tempo pode tornar o aplicativo inviável, fazendo com que os usuários desistam de utilizá-lo pelo tempo de resposta exigido para finalizar a aplicação do algoritmo, porém também é necessário que não seja um algoritmo tão simples, que possa ser facilmente quebrado ou decodificado, o que criaria brechas de segurança a fim de permitir que os arquivos fossem modificados de acordo com a vontade de um usuário mal intencionado.

IV. RESULTADOS, DISCUSSÕES E LIMITAÇÕES

Prontamente, cada abordagem projeta abertura para várias discussões relacionadas a seus resultados e limitações. Limitações que trabalhos futuros bem podem esplandecer. Tais resultados podem ser notados a seguir.

A. Mapeamento de dados de maior valor

A pesquisa resulta na consolidação de um modelo sucinto com diretrizes definidas para classificação, rotulagem e manuseio das informações que podem ser adotados dentro de um processo de ponto eletrônico.

Diretrizes:

- As pessoas somente devem possuir acesso às informações que sejam necessárias, direta ou indiretamente, ao desenvolvimento de suas

atividades de trabalho e demais responsabilidades associadas.

- A classificação da informação deverá ser realizada pelos gestores de cada área ou colaboradores designados por eles.
- O gestor deve garantir que todas as informações que compete a ele e sua área, esteja devidamente classificado e rotulado.
- Todos os colaboradores são responsáveis por tratar as informações de acordo com o seu nível de classificação, seguindo as diretrizes de tratamento aqui estipuladas.
- Periodicamente poderá ser realizado um processo de auditoria, monitoramento e medição para verificar a aderência no processo de classificação e tratamento da informação, a modo de obter métricas, sugestão de melhoria e para criar plano de ação.

Nível de classificação:

Níveis de Classificação	Características
Pública	Informações que podem ou devem ser divulgadas publicamente. A divulgação deste tipo de informação não causa problemas a Instituição ou a seus clientes, funcionários e parceiros, podendo ser compartilhada livremente com o público em geral, desde que seja mantida sua integridade.
Interna	Informações internas são aquelas divulgadas a todos os colaboradores e prestadores de serviços, desde que estes estejam comprometidos com a confidencialidade das informações.
Reservada	Informações confidencial são aquelas restritas a um determinado grupo, área ou cargo, que necessitem conhecê-las para o desempenho de suas tarefas profissionais na instituição.
Secreta/Confidencial	Informações Secretas/Confidenciais são aquelas que requerem um tratamento especial, pois cuja divulgação não autorizada ou acesso indevido, pode gerar prejuízos financeiros, legais, normativos, contratuais ou na reputação, imagem ou estratégia da Instituição.

Fig. 2 Nível de classificação [Imagem do autor]

Rotulação da informação

- Todas as informações da Instituição devem ser rotuladas com no mínimo o nível de classificação atribuída, grupo de acesso e data de criação;
- A rotulação da informação deve estar clara o nível de classificação e o grupo de acesso.
- A extração de sistemas de informações, deve conter rótulos apropriados da classificação da informação.
- O responsável pela rotulação da informação é o mesmo que fez o processo de classificação.
- Informações armazenadas em servidores de arquivos e outros dispositivos de armazenamento, devem possuir rótulo claro de grupo de acesso, sendo pasta compartilhada deve possuir bloqueio para acesso não autorizado.
- Os documentos em meios digitais devem possuir cabeçalho/rodapés, informando a categoria da informação.

Tratamento da Informação

- O tratamento adequado da informação deve prover maior controle e proteção, visando garantir sua confidencialidade, integridade e disponibilidade;
- O tratamento, cuidado e zelo com a informação, deve ser igual a todos, independentemente da pessoa, cargo ou área, para as mais diversas situações.

Segue uma figura referencial das diretrizes para o tratamento das informações em função do tipo de cenário e do nível de classificação.

Cenário	Público	Interno	Reservado	Secreto/confidencial
Acesso Lógico ou Físico	Sem Restrições	Somente para colaboradores da Instituição	Somente pessoas do grupo de acesso	Somente pessoas do grupo de acesso. Adicionalmente devem ser consideradas técnicas de proteção e integridade.
Armazenamento em arquivos digitais (rede)	Sem Restrições	Somente nos servidores de arquivos na rede da Instituição	Somente nos servidores de arquivos na rede da Instituição e com controle de acesso.	Somente nos servidores de arquivos na rede da Instituição e com controle de acesso. Preferencialmente com mais um nível de acesso (ex. criptografia).
Reprodução (impressa ou digital)	Sem Restrições	Somente para os colaboradores da Instituição	Somente para o grupo de acesso ou para outras pessoas somente com autorização do gestor responsável.	Somente com a autorização do gestor responsável.
Transmissão Digital	Sem Restrições	Interno, sem restrições. Para fora da Instituição, é necessário a autorização do gestor responsável.	Somente para o grupo de acesso. Para fora do grupo de acesso, é necessária a autorização do proprietário da informação.	Somente para o grupo de acesso. Para fora do grupo de acesso, é necessária a autorização do proprietário da informação. Adicionalmente devem ser consideradas técnicas de proteção e garantias de integridade.
Eliminação de mídia digital e/ou analógica	Sem Restrições	Somente dentro das áreas da Instituição	O dispositivo deverá ser destruído fisicamente ou as informações devem ser destruídas, apagadas ou sobre gravadas por meio de técnicas que tornem as informações originais irrecuperáveis.	O dispositivo deverá ser destruído fisicamente ou as informações devem ser destruídas, apagadas ou sobre gravadas por meio de técnicas que tornem as informações originais irrecuperáveis.

Fig. 3 Tratamento da Informação [Imagem do autor]

A classificação da informação deverá ser realizada pelos gestores de cada área, cabe aos profissionais da Segurança da Informação garantir que os requisitos de cada nível sejam devidamente atendidos. A efeito de exemplo, classifica-se os dados adquiridos pelo aplicativo de ponto eletrônico como secreto ou confidencial, visto que podem conter informações privadas dos colaboradores. Uma vez classificado com este nível devem-se considerar técnicas para garantir confidencialidade e integridade dos dados, tais como, medidas de segurança para o aplicativo, na transferência e recebimento destes dados. Assim como um diferente tipo de tratamento para cada cenário.

Uma vez que informação são recursos estratégicos para as organizações, em sua maioria é comum o não compartilhamento da forma ou metodologia que utilizam para classificação, rotulagem e tratamento de seus dados, os

quais poderiam contribuir para o enriquecimento do modelo proposto.

Um próximo passo seria a implementação deste modelo em todo o processo, em conjunto com os responsáveis por cada etapa ou setor, com isto definir de forma mais assertiva o valor de cada informação, bem como, a inclusão de cenários de acordo com a necessidade da instituição.

B. Detecção de malware com machine learning

Ao final do treinamento com a utilização do conjunto de dados do projeto "DroidFusion" [18], sob a perspectiva do algoritmo de classificação K-NN, pôde-se obter 98,33% de acurácia. O que evidentemente prova a capacidade de tal aplicação em detectar um *malware* em um dispositivo móvel cujo sistema operacional seja o Android. Tal acurácia, foi amplamente testada com várias métricas possíveis, além da seleção de vários valores de K.

Mantendo fixo a métrica da distância euclidiana, a figura a seguir mostra no eixo das ordinárias o valor da acurácia e no eixo das abscissas valores de K vizinhos.

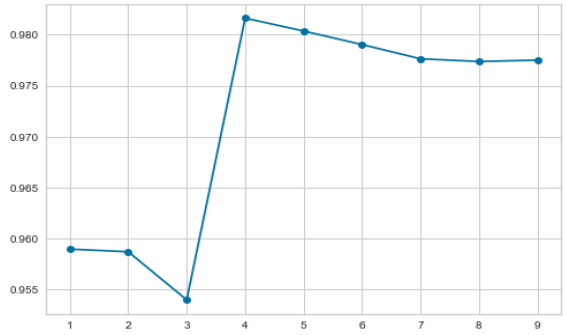

Fig. 4 Gráfico da acurácia pelos valores de K testados [Imagem do autor]

A imagem prova que quando K é igual a 4 o melhor resultado de acurácia é retornado. Percebe-se também que, para K menores que 4, a acurácia é consideravelmente menor. Por fim, nota-se que, para K maiores que 4, o aumento do valor de K reflete na acurácia diminuindo-a.

Outro teste importante foi o de manter fixo o K igual a 4 alterando apenas possíveis métricas de distâncias. Portanto, na figura a seguir o eixo das ordenadas continuam referindo-se ao valor de acurácia, entretanto o eixo das abscissas mostra agora as métricas possíveis.

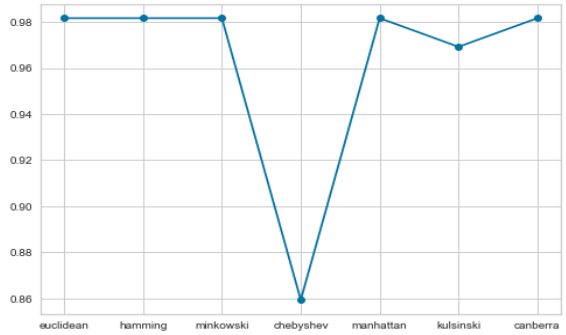

Fig. 5 Gráfico da acurácia pelas métricas testadas [Imagem do autor]

O resultado prova mais uma vez que a distância euclidiana, de fato, retorna acurácia máxima. Enquanto a métrica "Chebyshev" diminui consideravelmente o valor de acurácia retornado. Outro destaque negativo, porém, menos que o anterior, é o do "Kulsinski" que também retornou acurácia inferior à da distância euclidiana.

Vários são os artifícios estatísticos que podem medir o quão bom foram os resultados retornados após treinamento com o algoritmo K-NN. Um deles é a matriz de confusão que pode ser visualizada na imagem a seguir, onde a letra "B" significa resultados classificados de maneira supervisionada como "benigno" e a letra "S" as classificadas como "malware".

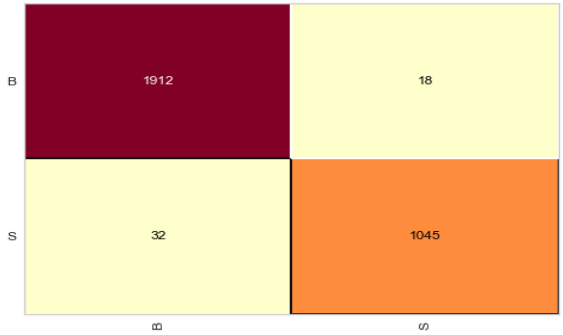

Fig. 6 Matriz de confusão relacionada as duas possíveis classificações [Imagem do autor]

Pode-se perceber na diagonal principal que os acertos foram amplamente superiores aos erros. Quando um conjunto de dados apontava para um caso verdadeiramente benigno, o algoritmo acertou 1912 vezes e errou apenas 32. Já quando o conjunto apontava para um caso, de fato, relacionado à malware, o algoritmo errou somente 18 vezes e acertou em 1045 situações.

Outra confirmação da satisfatoriedade do algoritmo aplicado, encontra-se na "*Classification Report*", outro artifício muito utilizado da biblioteca Scikit-learn.

	precision	recall	f1-score	support
B	0.98	0.99	0.99	1930
S	0.98	0.97	0.98	1077
accuracy			0.98	3007
macro avg	0.98	0.98	0.98	3007
weighted avg	0.98	0.98	0.98	3007

Fig. 7 Classification Report após treinamento do algoritmo [Imagem do autor]

Percebe-se que todas porcentagens sempre superam os 97%. Verifica-se aqui a alta precisão com que o algoritmo identificou cada linha do conjunto de dados.

Destaca-se que mesmo com alta precisão, o algoritmo atingiu um aprendizado que podemos considerar como "*Optimal*". Isso implica que o resultado não compactua com a situação de "*Underfit*" ou "*Overfit*", os altos resultados de acurácia e precisão se devem principalmente ao fato de o conjunto de dados passar por sumarização e normalização prévia.

Entretanto, é eminente a presença de algumas limitações neste projeto. O custo computacional, por exemplo, é uma clara limitação. Seria potencialmente arriscado aplicar um algoritmo tão oneroso com relação à memória em um dispositivo móvel que muitas das vezes tem foco em baixo custo e não em capacidade de *hardware*. A necessidade de supervisão também é uma limitação, já que para o algoritmo possa retornar cada vez melhores resultados, com o tempo, será necessário upgrade no conjunto de dados do treinamento. Além disso, o fato de os atributos do conjunto serem tão específicos, a normalização de maneira integra e automática precisaria adquiria maior viabilidade para que a usabilidade do algoritmo não seja infringida pelo próprio dispositivo. Para trabalhos futuros é interessante a proposta de otimizar a detecção de malware sob o uso de bibliotecas voltadas especificamente a desempenho e não a testes, como as utilizadas neste trabalho.

C. Criptografia para segurança na comunicação

Ao termino da elaboração do módulo de testes para comunicação segura pela internet, através da criptografia com uso do protocolo Diffie Hellman para geração de chaves de assinatura e criptografia dos dados, obteve-se com sucesso um fluxo de operação que agregue confidencialidade e autenticidade aos dados transmitidos.

Todavia o trabalho foi baseado em uma biblioteca importada, a qual não apresenta argumentos de que as funções elaboradas na biblioteca seguem as recomendações e modelos apresentados em pesquisas e trabalhos de organizações de renome, como: NIST, IETF e IEEE, de modo que o modelo ideal se dá com desenvolvimento de uma biblioteca própria a qual é possível garantir o cumprimento das recomendações feitas por tais organizações.

Outra limitação discutida se dá com a execução do algoritmo em um ambiente que não simule as condições reais ao qual será implementado, referindo ao *smartphone* do colaborador que irá utilizar do ponto eletrônico digital. É previsto que tal dispositivo possua limitações de recursos e processamento, sendo esse um dos motivos ao qual a curva 25519 foi alvo de estudo e implementada.

E por fim outro ponto limitante é a ausência de um sistema completo de ponto digital, com aplicativo de celular e servidor central para realização de testes com troca de informações via internet em paralelo a execução dos demais módulos de segurança propostos.

D. Hashing de verificação

Utilizando-se de um banco de dados para testes, aplicou-se através da biblioteca hashlib [44] as funções *hash*: md5, sha1, sha256 e sha512 nas informações presentes em cada um dos registros, gerando respectivamente saídas de: 128 bits, expresso em 32 caracteres; 160 bits, expressa em 40 caracteres; 256 bits, expressa em 64 caracteres; 512 bits, expressa em 128 caracteres. Com a biblioteca timeit [45], pôde-se verificar o tempo que cada uma dessas funções levou para ser aplicada no banco de dados de testes, obtendo um resultado de 0,261s para a função *hash* MD5, 0,235s para a função *hash* SHA1, 0,240s para a função *hash* SHA256 e 0,301s para a função *hash* SHA512. Apesar de pouca diferença em um banco de dados pequeno, nota-se que a função *hash* MD5, apesar de ser mais simples que as *security hash algorithm* (SHA), apresentou um tempo de execução maior que a SHA-1 e a SHA-256.

Com base nos resultados de tempo obtidos mediante os testes com a biblioteca timeit, selecionou-se a função *hash* SHA256 para ser aplicada no banco de dados, uma vez que a diferença de tempo de execução em relação à MD5 e SHA1 é pouca, e as vantagens de segurança da função SHA256 são maiores, pois esta oferece maior resistência à colisão, e ataques à pré-imagem e à segunda pré-imagem [46]. Através do uso da biblioteca hashlib, pôde-se elaborar o algoritmo que realizava a aplicação da função *hash* SHA256 nos dados gerados pelo aplicativo de controle de ponto, e constatar-se que, de acordo com o esperado, qualquer mínima alteração realizada em um dos campos do banco de dados da cópia faz com que a *hash* gerada seja completamente diferente da original, fato esse que é evidenciado quando se comparam as funções *hashes* da tabela original com a da cópia, que pode ou não ter sido modificada.

Através desta comparação, têm-se então um sistema de verificação de integridade dos arquivos, entretanto, é válido ressaltar que esta abordagem não permite que se identifique quais dados foram alterados, pois a *hash* não pode ser revertida para os dados originais, nem identificar o responsável por tais alterações, pode-se somente realizar a identificação de manipulação, atrelando essa informação a um ID de registro, também presente no banco de dados, para que então seja conferido de forma manual os campos que podem ter sido manipulados.

V. CONCLUSÕES

Todos os dados que trafegam da origem até o destino do processo de ponto eletrônico deverão ser mapeados e atribuído um nível adequado. Dessa forma pode-se mensurar o impacto e a prioridade em realizar sua segurança. O modelo de segurança da informação aqui estabelecido, define diretrizes para devida classificação, rotulagem e manuseio das informações, acordo com a sua sensibilidade para a instituição. A classificação de dados não somente serve para justificar as medidas de proteção implementadas, mas também para a conscientização de todas as partes envolvidas no processo como um todo, podendo ser aplicada em qualquer setor, departamento ou instituição como parte de sua política de segurança.

Faz-se necessário que o ambiente no qual a aplicação encontra-se em execução esteja seguro, portanto a detecção de malware no dispositivo móvel do colaborador possui elevado valor. Garantir a segurança da aplicação diretamente na fonte, considerando um Android, passa muito por analisar os dados de chamada de API e permissões do arquivo de manifesto do sistema. Na análise torna-se viável a aplicação do algoritmo inteligente, que neste trabalho foi o K-NN, a fim de classificar o conjunto de dados como benigno ou como infectado por malware. Dada a acurácia significativa de mais de 98% que o algoritmo retornou, tem-se que que a abordagem de *machine learning* escolhida retorna, de fato, desempenho *optimal*.

A implementação de um sistema que garanta a confidencialidade junto a autenticidade dos dados providos do aplicativo se faz como um quesito importante, uma vez que nos dias atuais existem diversos métodos que monitoram transferências de dados em meios públicos, referente a transição pela internet ilustrada no *Threat Model*. Mesmo diante das diversas limitações listadas no capítulo anterior o módulo apresentou resultados satisfatórios que podem ser

melhor desenvolvidos à medida que tais limitações sejam superadas ou pivotadas.

Através da utilização de um algoritmo que aplica uma função hash em um banco de dados, pôde-se criar um modelo seguindo o *Threat Model*, disposto na figura 1 que tornasse possível realizar a verificação de integridade dos dados dispostos neste banco de dados. Além disso, foi realizada uma breve verificação de qual algoritmo melhor se aplicava a fim de não causar prejuízos à disponibilidade da aplicação. De tal forma, têm-se maior controle sobre os dados que foram modificados, mitigando assim prejuízos que pudessem ser ocasionados por essa brecha na segurança das informações, enquanto estas estão sendo submetidas ao processo de verificação e validação.

Não obstante, garantir o ambiente seguro é de extrema importância ao detectar *malware* nos dispositivos móveis. Grande valor também tem a utilização do protocolo *Diffie-Hellman* junto aos métodos aplicados ao compartilhamento, geração e eliminação de chaves de sessão dependendo da fase onde se encontra o fluxo da aplicação. Por fim, no tocante à integridade dos dados gerados pelos usuários relacionados com possíveis alterações no meio do processo, o módulo de segurança que abordou o uso de *hashing* bem permite que a integridade seja verificada de forma efetiva. Dada a integração de todas essas abordagens de segurança cibernética para aplicativos de ponto eletrônico, apesar das evidentes limitações, pode-se afirmar com assertividade que, em medida aceitável, os pilares da segurança cibernética foram contemplados.

RECONHECIMENTO

Os autores manifestam aqui os sinceros agradecimentos àqueles que grande contribuição deu ao conhecimento interdisciplinar necessário para o desenvolvimento deste trabalho. Somos eternamente gratos ao Professor Mestre Nilton Stringasci Moreira, ao Professor Mestre Rodrigo Hiroshi Ruiz Suzuki, ao Professor Pós-doutor João Augusto Mattar Neto, ao Professor Mestre Osmany Dantas Ribeiro de Arruda, ao Professor Mestre Paulo Rogerio Nietto e à Professora Mestra Andréia Damasio Leles que até aqui compartilharam experiências conosco no Centro Universitário FACENS a fim de obtermos notórios desempenhos na Especialização em Segurança Cibernética em parceria com a LENOVO.

REFERÊNCIAS

[1] "Controle de ponto para funcionários externos". PontoTel. https://www.pontotel.com.br/controle-de-ponto-para-funcionarios-externos/ (consult. 2022-11-04).

[2] L. Barros. "Qual o melhor app de controle de ponto?" Tangerino - Sistema de Controle de Ponto Eletrônico. https://tangerino.com.br/blog/controle-de-funcionario-externo-com-app-de-controle-de-ponto/ (consult. 2022-11-04).

[3] "Entenda qual é a diferença entre dado e informação - Inovação - Sebrae". Inovação - Sebrae. https://inovacaosebraeminas.com.br/entenda-qual-e-a-diferenca-entre-dado-e-informacao/#:~:text=Independentemente%20de%20setores%20ou%20de,forma%20incorreta,%20costumam%20se%20misturar. (consult. 2022-11-04).

[4] Tecnologia da informação - Código de prática para a gestão da segurança da informação, NBR ISO/IEC 17999:2005, ABNT - Associação Brasileira de Letras, Rio de Janeiro, 2001.

[5] M. Araujo. "Tenha em mãos os principais documentos do setor de RH". eBox Digital. https://www.eboxdigital.com.br/blog/tenha-em-maos-os-principais-documentos-do-setor-de-rh (consult. 2022-11-04).

[6] Tecnologia da Informação-Técnicas de Segurança – Código de Prática para controles de segurança da informação, NBR ISO/IEC 27002:2013, ABNT - Associação Brasileira de Letras, Rio de Janeiro, 2013.

[7] L. S. Mendes, "Transparência e privacidade : Violação e proteção da informação pessoal na sociedade de consumo", publishedVersion, reponame:Repositório Institucional da UnB, 2008. Consult. 2022-11-05. [Em linha]. Disponível: http://repositorio.unb.br/handle/10482/4782

[8] D. Santos e N. Rodrigues, "Política de armazenamento, anonimização e descarte", ÁLAMO ENGENHARIA S/A, São Paulo/SP, P(LGPD)07, junho de 2021. Consult. 2022-11-01. [Em linha]. Disponível: https://alamoengenharia.com.br/wp-content/uploads/2021/05/PLGPD07-Politica-de-Armazenamento-Anonimizacao-e-Descarte-Rev-01.pdf

[9] B. Reis, J. Mota e P. Oliveira. "Classificação da informação". Universidade Católica de Brasília (UCB), Campos Universitário II, SGAN 916 – Módulo B – Brasília – DF – Brasil. http://www.lyfreitas.com.br/ant/artigos_mba/artclassinfo.pdf (consult. 2022-11-01).

[10] Souppaya, Murugiah e Karen Scarfone: Guide to malware incident prevention and handling for desktops and laptops, 2013. 8, 11, 14, 17, 34.

[11] CERT.br: Códigos maliciosos (malware). Disponível em: . [Acesso em 08 set 2020 do Gelson], 2017. 9 .

[12] É. A. Barros, "Vírus de computadores: uma abordagem histórica e prática", 2003.

[13] Kaur, Simarleen e Arvinder Kaur: Detection of Malware of Code Clone using String Pattern Back Propagation Neural Network Algorithm. Indian Journal of Science and Technology. doi: 10.17485/ijst/2016/v9i33/95880, 9(33), setembro 2016, ISSN 0974-5645, 0974-6846. 9, 12 .

[14] Barbara, Santa, Clemens Kolbitsch, Paolo Milani Comparetti e Ludovico Cavedon: (54) Methods and Systemis Formalware Detection Based on Environment Dependent behavior. página 12, 2016. 10.

[15] Sihwail, Rami, Khairuddin Omar e Khairul Akram Zainol Ariffin: A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis. International Journal on Advanced Science, Engineering and Information Technology, arXiv:1710.09435, 8:1662–1671, 2018, ISSN 2088-5334. 9, 10, 11, 12, 13.

[16] L. P. d. Melo, D. M. Amaral, F. Sakakibara, A. R. d. Almeida, R. T. d. S. Junior e A. Nascimento, "Análise de Malware: Investigação de Códigos Maliciosos Através de uma Abordagem Prática", Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, 2002.

[17] R. C. Gomes et al., "Sistema Operacional Android", Universidade Federal Fluminense, 2012. S. Y. Yerima and S. Sezer, ‘‘DroidFusion: A novel multilevel classifier fusion approach for Android malware detection,’’ IEEE Trans. Cybern., vol. 49, no. 2, pp. 453–466, Feb. 2018.

[18] S. Y. Yerima and S. Sezer, ‘‘DroidFusion: A novel multilevel classifier fusion approach for Android malware detection,’’ IEEE Trans. Cybern., vol. 49, no. 2, pp. 453–466, Feb. 2018.

[19] Y. Zhou and X. Jiang, ”Dissecting android malware: Characterization and evolution” In proc. 2012 IEEE Symposium on Security and Privacy (SP), San Fransisco, CA, USA, 20-23 May, 2012 , pp. 95-109.

[20] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, and K. Rieck, ”Drebin: Efficient and Explainable Detection of Android Malware in Your Pocket” In proc. 20th Annual Network & Distributed System Security Symposium (NDSS), San Diego, CA, USA, 23-26 Feb. 2014.

[21] Y. Aafer, W. Du, and H. Yin, ”DroidAPIMiner: Mining API-level features for robust malware detection in Android” In proc. 9th

Int.Conference on Security and Privacy in Communication Networks (SecureComm 2013). Sydney, Australia, Sep. 25-27, 2013.

[22] T. Book, A. Pridgen, and D. S. Wallach, ”Longitudinal Analysis of Android Ad Library Permissions” In proc. Mobile Security Technologies conference (MoST 13), San Fransisco, CA, May 2013.

[23] D. Stiawan, S. M. Daely, A. Heryanto, N. Afifah, M. Y. Idris e R. Budiarto, "RANSOMWARE Detection Based on Opcode Behaviour Using K-Nearest Neighbours Algorithm", Information Technology and Control, vol. 50, n.º 3, pp. 495–506, 2021-09-24. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.5755/j01.itc.50.3.25816 .

[24] Y. Wang, Z. Pan e J. Dong, "A new two-layer nearest neighbor selection method for K-NN classifier", Knowledge-Based Systems, vol. 235, n.º 107604, 2022-01-10. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.1016/j.knosys.2021.107604 .

[25] A. Oztekin, D. Delen, A. Turkyilmaz e S. Zaim, "A machine learning-based usability evaluation method for eLearning systems", Decision Support Systems, vol. 56, pp. 63–73, 2013-12-01. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.1016/j.dss.2013.05.003

[26] H. U. Ullah et al., "Comparative study for machine learning classifier recommendation to predict political affiliation based on online reviews", CAAI Transactions on Intelligence Technology, vol. 6, n.º 3, pp. 251–264, 2021-05-07. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.1049/cit2.12046 .

[27] D. De Freitas Aranha e R. Junio Da Cruz, "Análise de segurança em aplicativos bancários na plataforma android", in XXIII congresso de iniciação científica da unicamp, 2015-11-17–19. Campinas - SP, Brazil: Galoá, 2015. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.19146/pibic-2015-37751 .

[28] R. M. Pires e V. G. Costa, "Uso do algoritmo Diffie-Hellman na geração de keyfile para o TrueCrypt", Encontro Anual de Computação (ENACOMP), vol. VIII, janeiro de 2010. Consult. 2022-11-04. [Em linha]. Disponível: https://www.researchgate.net/publication/313216578 .

[29] D. J. Bernstein, "Curve25519: New diffie-hellman speed records", in Public Key Cryptography - PKC 2006. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 207–228. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.1007/11745853_14.

[30] A. C. Guimarães, E. Borin e D. F. Aranha, "Extensão do conjunto de instruções para implementação segura de X25519", in SBSeg16 - XVI simpósio brasileiro em segurança da informação e de sistemas computacionais, Niteroi - RJ, Brasil, 2016-11-07–10. Porto Alegre - RS: SBC — Soc. Bras. de Computação, 2016, pp. 625–635. Consult. 2022-11-04. [Em linha]. Disponível: https://sbseg2016.ic.uff.br/pt/files/anais/wticg/ST4-1.pdf .

[31] D. Brumley e D. Boneh, "Remote timing attacks are practical", Computer Networks, vol. 48, n.º 5, pp. 701–716, agosto de 2005. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.1016/j.comnet.2005.01.010 .

[32] I. Antunes e L. A. Kowada, "Explorando o sistema de criptografia signal no whatsapp", in SBSeg18 - XVIII simpósio brasileiro de segurança da informação e de sistemas computacionais, Natal - RN, Brasil, 2018-10-22–25. Porto Alegre - RS: SBC — Soc. Bras. de Computação, 2018, pp. 181–195. Consult. 2022-11-04. [Em linha]. Disponível: https://sol.sbc.org.br/index.php/sbseg/article/view/4252.

[33] H. Krawczyk e P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC Editor, maio de 2010. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.17487/rfc5869

[34] "Advanced encryption standard (AES)", National Institute of Standards and Technology, Gaithersburg, MD, novembro de 2001. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.6028/nist.fips.197.

[35] D. A. McGrew e J. Viega, "The security and performance of the galois/counter mode (GCM) of operation", in Progress in Cryptology - INDOCRYPT 2004. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 343–355. Consult. 2022-11-04. [Em linha]. Disponível: https://doi.org/10.1007/978-3-540-30556-9_27 .

[36] L. Malina, J. Hajny e V. Zeman, "Usability of pairing-based cryptography on smartphones", in 2015 38th International Conference on Telecommunications and Signal Processing (TSP), Prague, Czech Republic, 2015-07-09–11. IEEE, 2015. Consult. 2022-11-05. [Em linha]. Disponível: https://doi.org/10.1109/tsp.2015.7296337 .

[37] F. Turan e I. Verbauwhede, "Compact and Flexible FPGA Implementation of Ed25519 and X25519", ACM Transactions on Embedded Computing Systems, vol. 18, n.º 3, pp. 1–21, junho de 2019. Consult. 2022-11-05. [Em linha]. Disponível: https://doi.org/10.1145/3312742 .

[38] S. Josefsson e I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC Editor, janeiro de 2017. Consult. 2022-11-05. [Em linha]. Disponível: https://doi.org/10.17487/rfc8032 .

[39] Tecnologia da informação - Código de prática para a gestão da segurança da informação, NBR ISO/IEC 17799, ABNT – Associação Brasileira de Normas Técnicas, Rio de Janeiro, 2001.

[40] M. Singh and D. Garg, "Choosing Best Hashing Strategies and Hash Functions," 2009 IEEE International Advance Computing Conference, 2009, pp. 50-55, doi: 10.1109/IADCC.2009.4808979.

[41] E. Demaine, S. Devadas e N. Lynch. "Lecture 21: Cryptography: Hashing". MIT OpenCourseWare | Free Online Course Materials. Massachusetts Institute of Technology: MIT OpenCourseWare. Licence: Creative Commons BY-NC-SA. https://ocw.mit.edu/courses/6-046j-design-and-analysis-of-algorithms-spring-2015/6741d65be662edac7c169c4081b3bd9a_MIT6_046JS15_lec21.pdf (consult. 2022-11-04).

[42] A. Dolinay. "Cryptography Workshops - Cryptographic Hashing". GitHub. https://github.com/tudev/Workshops-2020-2021/blob/master/Cryptography%20Workshops/TUDev's_Cryptography_Workshop!_Workshop_II_Cryptographic_Hashing_(FULL).ipynb (consult. 2022-11-04).

[43] C. Henrique Calazans Ribeiro, A. Yuuji Hira e M. Knörich Zuffo. "Aplicação da Técnica de Duplo Hash na Implementação de Serviços de Integridade em Registros Eletrônicos de Saúde". Disponível: https://www.researchgate.net/profile/Marcelo-Zuffo/publication/266606943_Aplicacao_da_Tecnica_de_Duplo_Hash_na_Implementacao_de_Servicos_de_Integridade_em_Registros_Eletronicos_de_Saude/links/545bb1890cf2f1dbcbcafff4/Aplicacao-da-Tecnica-de-Duplo-Hash-na-Implementacao-de-Servicos-de-Integridade-em-Registros-Eletronicos-de-Saude.pdf (consult. 2022-11-04).

[44] "hashlib — Secure hashes and message digests". Python 3.11.0 documentation. https://docs.python.org/3/library/hashlib.html (consult. 2022-11-04).

[45] "timeit — Measure execution time of small code snippets". Python 3.11.0 documentation. https://docs.python.org/3/library/timeit.html.

[46] "The Ultimate Hash Algorithm Comparison: MD5 vs. SHA-1 vs. SHA-2 vs. SHA-3". Hash Algorithm Comparison: MD5, SHA-1, SHA-2 & SHA-3. https://codesigningstore.com/hash-algorithm-comparison (consult. 2022-11-04).