Revisitando um Protocolo para Acesso a Redes Sem Fio Públicas e Gratuitas

Danilo D. P. Paraiso[1]

[1] Instituto de Ciências Exatas e Naturais
Universidade Federal do Pará (UFPA) – Belém, PA – Brasil

danilo.paraiso@icen.ufpa.br

Abstract. *Public and free wireless networks have become commonplace and are an important tool for social inclusion. Such networks allow free access to the Internet through mobile devices. Although they provide countless benefits to their users, they can be used to commit crimes such as drug trafficking. In a previous work, a solution was proposed that allows the identification of crimes in these networks. This solution, however, has not been implemented and has some disadvantages. This work revisits its own through its implementation and introducing improvements.*

Resumo. *Redes sem fio públicas e gratuitas tornaram-se comuns e são uma importante ferramenta de inclusão social. Tais redes possibilitam livre acesso à Internet por meio de dispositivos móveis. Embora forneçam inúmeros benefícios a seus usuários, elas podem ser utilizadas para a prática de crimes virtuais. Em um trabalho anterior, foi proposta uma solução que permite a identificação de crimes nessas redes. Essa solução, no entanto, não foi implementada e possui algumas desvantagens. Este trabalho revisita a própria por meio de sua implementação e introduzindo melhorias.*

1. Introdução

Redes sem fio públicas e gratuitas tornaram-se comuns e são uma importante ferramenta de inclusão social. Elas possibilitam o acesso à Internet em locais públicos, por qualquer um que disponha de um dispositivo móvel como *smartphones* e *tablets*. A liberdade de acesso à Internet, no entanto, traz diversos riscos associados como a utilização dessas redes para o cometimento de atividades ilícitas como roubo e vazamentos de dados. Tais atividades podem resultar em fraudes financeiras e danos à privacidade dos demais usuários [Zimmer 2022].

[Lucena 2022] destaca que, nos últimos anos, os ambientes que passaram a oferecer Wi-Fi público cresceram, porém têm se mostrado expressivamente inseguros e muitos já são os casos de crimes virtuais cometidos por meio da rede sem fio, sem que a polícia conseguisse chegar até o responsável pela ação ilegal.

Uma das maneiras de se atenuar atividades ilícitas em redes Wi-Fi públicas é por meio da coleta de dados dos usuários. Esses dados podem ser usados para identificar e rastrear usuários maliciosos, limitando suas ações. Por exemplo, ao coletar dados dos dispositivos que se conectam à rede, é possível detectar dispositivos não autorizados e impedi-los de acessar a rede. Da mesma forma, monitorar a atividade do usuário pode ajudar a identificar comportamentos suspeitos e prevenir possíveis ataques. No entanto,

a coleta de dados deve ser realizada de acordo com a Lei Geral de Proteção de Dados (LGPD).

A proposta de [MELO and ARAUJO 2021] é o objeto de estudo deste trabalho, pois originou o protocolo cujo cerne consistia em associar o usuário ao seu próprio dispositivo móvel (e.g. *smartphone*) quando conectado às redes Wi-Fi. Com essa intenção, esperava-se que crimes virtuais fossem reduzidos e possivelmente até evitados, pois, uma vez relacionado o usuário ao seu *smartphone*, sua identificação tornar-se-ia mais fácil. No entanto, tal protocolo não foi implementado e necessitava de melhorias. Logo, o trabalho a seguir implementa o protocolo anterior, adicionando novas tecnologias e componentes, como o desenvolvimento de uma aplicação *mobile* (interface para o usuário), o *Jason Web Token* (JWT) e um novo identificador único para dispositivos android.

Este trabalho está organizado da seguinte forma. A Seção 2 apresenta os conceitos e tecnologias utilizadas aqui. Em seguida, a Seção 3 apresenta o protocolo objeto de estudo desse trabalho. A Seção 3 revisita o protocolo anterior introduzindo e discutindo melhorias. Por fim, a Seção 5 conclui este trabalho e apresenta os trabalhos futuros.

2. Fundamentação Teórica

O trabalho introduzido aqui utiliza-se de algumas tecnologias e mecanismos criptográficos conhecidos da literatura. Além disso, visto que envolve a obtenção de dados dos usuários, ele tem relação com algumas Leis brasileiras relacionadas à proteção desses dados. A seguir, tais mecanismos, bem como as Leis, serão apresentados.

2.1. Função Hash Criptográfica

Um função hash criptográfica calcula um identificador digital de tamanho fixo, chamado de valor hash, a partir de uma entrada de tamanho variável. Ou seja, para uma mensagem de tamanho m, ela produz o valor de hash de tamanho fixo $h = \mathrm{H}(m)$ [Stallings 2014].

A função criptográfica de *hash* considerada segura tem como principais propriedades: (1) ser unidirecional, ou seja, deve ser inviável obter a mensagem a partir do valor valor hash; (2) resistência à segunda pré-imagem, isto é, não devem existir dois valores de entrada com a mesma saída. (3) resistente à colisão, isso significa que deve ser computacionalmente inviável encontrar duas mensagens que tenham o mesmo valor de hash (colisão).

Uma função hash criptográfica pode ser comparada a um dígito verificador, ou até mesmo a um algoritmo de controle, promovendo um mecanismo que pode assegurar a integridade das respectivas informações.

2.2. Criptografia Assimétrica, Assinatura Digital, e Certificados Digitais X.509

A criptografia assimétrica, também conhecida como criptografia de chave pública, é regida por duas chaves relacionadas, uma pública e uma privada. Elas são utilizadas para realizar operações complementares, como encriptação e decriptação ou geração e verificação de assinatura. Se uma chave for utilizada para encriptação, a outra é usada para decriptação. Ambas compartilham a propriedade de que deve ser computacionalmente inviável a derivação da chave privada a partir da pública. O algoritmo que gera essas chaves pode ser utilizado também nas assinaturas digitais.

As assinaturas digitais, por sua vez, fornecem autenticidade, integridade e não repúdio aos dados. O esquema de assinatura digital dispõe da ação de três algoritmos: (1) o algoritmo de geração de chaves gera um uma chave privada e uma pública; (2) o algoritmo de geração de assinatura calcula uma assinatura utilizando a chave privada. (3) E o algoritmo de verificação de assinatura é fornecido junto ao dado que está sendo assinado (e.g., documento), além da própria assinatura e da chave pública que verificará se a assinatura é válida. Desse modo, se validada, a assinatura garante que nenhum dado foi alterado (integridade), identifica quem assinou (autenticidade) e impede que o autor da assinatura negue a si próprio (não repúdio).

Quanto ao certificado digital, ele é a identidade digital da pessoa física e jurídica no meio eletrônico. Ele garante autenticidade, confidencialidade, integridade e não repúdio nas operações que são realizadas por meio dele, atribuindo validade jurídica [Certisign 2023]. Ele é composto por: versão e número de série do certificado, identificação do órgão de controle (CA), identificação e chave pública do dono do certificado, validade do certificado e assinatura digital.

Por ser utilizado como identificação no meio digital, ele permite que diversos serviços sejam realizados sem a necessidade da presença física, o que significa agilidade nos processos, sustentabilidade e redução de custos. Sendo assim, pode-se definir o certificado digital como um registro eletrônico constituído por um conjunto de dados que identifica uma entidade, associando uma chave pública à própria. O certificado digital pode ser emitido para uma pessoa, empresas, instituições, equipamentos ou serviços na rede e pode ser comparado a um documento de CPF ou identidade, os quais possuem dados pessoais e intransferíveis.

2.3. O Protocolo TLS

Segundo o RFC 8446 [Dirks and Allen 1999], o protocolo TLS foi concebido para permitir a comunicação via Internet de forma segura, impedindo, assim, interceptação, alteração e falsificação de mensagens. O TLS utiliza uma combinação de criptografia simétrica e assimétrica para fornecer segurança. O algoritmo de criptografia simétrica é usado para criptografar os dados, enquanto que o algoritmo assimétrico é usado para estabelecer um canal seguro entre os terminais, além de também ser utilizado para autenticar as duas partes da comunicação, garantindo que os dados sejam transmitidos de forma confiável.

O protocolo TLS é o sucessor do protocolo *Secure Sockets Layer* (SSL) e passou por várias revisões desde a sua criação, sendo o TLS 1.2 a versão mais usada atualmente. O TLS 1.3 foi lançado em 2018, aprimorando a segurança e tornando *handshakes* (estabelecimento de conexão) mais rápidos.

É inegável a sua criticidade ao proteger a comunicação *online*. Com o aumento da quantidade de informações confidenciais sendo transmitidas pela Internet, ele é crucial para proteger tais informações de invasores que tentam manipular ou até roubar dados sigilosos. Como tal, é impreterível que sites e outros serviços online o implementem para garantir a segurança dos dados de seus usuários.

2.4. O Sistema Operacional Android

O sistema operacional Android começou a ser desenvolvido em 2005, quando o Google comprou a Android, Inc. Três anos depois ele lançou a primeira versão comercial e estável

do que anos mais tarde tornar-se-ia o sistema operacional mais utilizado no mundo. Por possuir código aberto, desenvolvedores podem criar suas próprias versões personalizadas do sistema, utilizando as linguaguens de programação Java e Kotlin. Ambas são definidas pelo Google como as principais no desenvolvimento de aplicações android.

O *kernel* do Android é baseado no *kernel* do sistema operacional Linux. Visto que o *kernel* do Linux é monolítico, o Android no começo herdou essa característica. Sua arquitetura possui quatro camadas, que são divididas entre: a de aplicações, onde ficam os aplicativos de usuário e a parte *front-end*; a de framework, onde ficam as APIs de desenvolvimento de aplicativos para o sistema; a de bibliotecas, que detém funcionalidades do sistema e a camada do próprio kernel do Linux [Android 2023].

Para se trabalhar com identificadores exclusivos que poderão ser utilizados nas aplicações, em [Developers 2023c] é possível consultar as orientações para selecionar tais identificadores com base no caso de uso. Um desses identificadores, portanto, foi utilizado para a implementação deste trabalho, o AndroidID [Developers 2023b]. Ele foi cogitado junto ao *IMEI*, no entanto o IMEI não pôde ser utilizado por maiores restrições de uso nas versões mais atuais do sistema [Developers 2023a].

Quanto ao AndroidID, ele é especificamente um número de 64 bits (expresso como uma string hexadecimal), exclusivo para cada combinação de chave de assinatura de aplicativo, usuário e dispositivo, tornando-se, dessa forma, um candidato adequado à função de identificador exclusivo para a proposta aqui apresentada.

2.5. JSON Web Token (JWT)

O JSON Web Token (JWT) é um padrão aberto (RFC 7519 [Jones et al. 2015]) que define uma maneira compacta e independente de transmitir informações com segurança entre as partes, como um objeto JSON. Tais informações são assinadas digitalmente por criptografia simétrica (e.g., o algoritmo HMAC) ou por criptografia assimétrica, por meio da chave privada (e.g., algoritmos RSA ou ECDSA).

O JWT é comumente utilizado para a troca de informações, principalmente entre *backends* diferentes, como no caso da arquitetura de microsserviços. Ele é uma alternativa ao *Session Storage* como autenticação do usuário, pois, diferentemente das sessões clássicas, o servidor não precisa armazenar a sessão do usuário no banco de dados, por exemplo. No caso do JWT, o servidor gera um par de chaves públicas e pode utilizar a chave privada para assinar os tokens de cada usuário. Após realizar o acesso via *login*, cada solicitação subsequente inclui o JWT assinado, permitindo que o usuário acesse rotas, serviços e recursos permitidos com esse token.

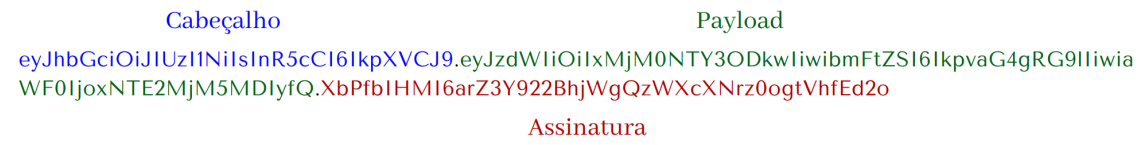

Figura 1. Representação de um Token JWT.

Em sua forma compacta (ver Figura 1), o JWT consiste em três partes: o cabeçalho (*header*), a carga útil (*payload*) e Assinatura. O cabeçalho em suma consiste em duas partes: o tipo do token, que é JWT, e o algoritmo de assinatura empregado, como HMAC

SHA256 ou RSA. A carga útil é composta por declarações/reivindicações (*claims*). Tais reivindicações são declarações sobre uma entidade (normalmente, o usuário) e dados adicionais (e.g., assunto). A assinatura objetiva garantir a autenticidade e a integridade da mensagem assinada. No caso de tokens assinados, é possível verificar se o remetente do JWT é quem diz ser.

Com relação à carga útil, existem três tipos de declarações: as registradas, as públicas e as privadas. As declarações registradas são um conjunto de declarações que não são obrigatórias, porém recomendadas para fornecer um conjunto de declarações úteis e interoperáveis, como *iss* (emissor), *exp* (tempo de expiração), e *sub* (assunto). Quanto às públicas, elas podem ser declaradas à vontade pelos usuários do JWT. Por fim, as declarações privadas são declarações personalizadas criadas para compartilhar informações entre as partes que concordam em usá-las e não são, portanto, registradas ou públicas.

2.6. O Marco Civil da Internet e a Lei Geral de Proteção de Dados Pessoais

A Lei Geral de Proteção de Dados Pessoais (LGPD)[Executivo 2018] tem por objetivo proteger os direitos fundamentais de liberdade e de privacidade e a livre formação da personalidade de cada indivíduo. Trata-se de um conjunto de regras sobre o tratamento de dados pessoais armazenados por parte do serviço da aplicação.

A LGPD aplica-se a todas as empresas e instituições que coletam, armazenam ou processam dados pessoais, sejam elas de caráter público ou privado. Ela estabelece normas claras acerca de como esses dados devem ser tratados e prevê sanções para as empresas que não cumprirem as regras. Tais regras visam proteger a captação, armazenamento e compartilhamento de dados pessoais coletados por sites e empresas. Ressalta-se que o primeiro avanço jurídico nesse sentido teve início em 2014, com o Marco Civil da Internet (MCI) [Executivo 2014], que buscou regular os direitos e deveres de um determinado indivíduo na rede.

O Marco Civil da Internet (MCI) estabelece princípios, garantias, direitos e deveres para o uso da Internet no país. Sancionada em 2014, a Lei tem como objetivo proteger a privacidade dos usuários, promover a liberdade de expressão e garantir a neutralidade da rede. O MCI é uma Lei fundamental que almeja garantir a liberdade de expressão, a privacidade dos usuários e a neutralidade da rede no Brasil.

A proposta aqui apresentada tem relação direta com a LGPD pois envolve a coleta de dados dos usuários. Dessa forma, seguindo a LGPD, é necessário que os usuários autorizem o uso e o armazenamento de seus dados.

3. Um Protocolo Para Liberação de Acesso Seguro a Redes Sem Fio Públicas Gratuitas

Em uma versão preliminar da proposta aqui apresentada, [MELO and ARAUJO 2021] introduziu um protocolo para liberação de acesso seguro em redes sem fio públicas. Tal protocolo é a base do trabalho proposto aqui. Esse protocolo é descrito a seguir.

O protocolo considera dois participantes. O usuário (ou cliente) que realiza o acesso à Internet e o servidor (ou servidor) que autoriza e provém o serviço de Internet aos clientes.

A fim de se obter acesso à Internet, o usuário primeiramente realiza um primeiro acesso ao servidor. Para isso, ele estabelece uma conexão TLS com o servidor (e.g., via navegador *Web*). Após essa conexão, o servidor envia a ele três códigos (i.e., três números aleatórios gerados para esse fim). Dois desses códigos são enviados utilizando diferentes canais de comunicação (e.g., *e-mail* e SMS). O terceiro código é enviado por meio da própria aplicação (e.g., via navegador *Web*). Ao receber esses códigos, o cliente utiliza um algoritmo de criptografia simétrica pré-definido (e.g., o algoritmo AES) e criptografa os códigos recebidos, o IMEI do dispositivo móvel e outros dados do usuário. Ele também calcula um código de autenticação de mensagem (MAC) (e.g., via algoritmos HMAC) e o envia junto ao texto cifrado para o servidor. Todo o processamento criptográfico é realizado localmente pela aplicação utilizando como chave simétrica a chave de sessão do protocolo TLS. Essa chave é definida previamente durante o estabelecimento da conexão TLS.

Ao receber tais dados, o servidor utiliza a chave de sessão TLS para decriptografá-los, verifica o MAC e se os códigos recebidos são os mesmos enviados. Posteriormente, ele gera um token de acesso temporário, o TTA. O TTA é calculado aplicando-se uma função hash criptográfica (e.g., SHA-256) que tem como entrada o IMEI do dispositivo, o número do *smartphone* e outros dados do usuário. Após a geração do TTA, ele é criptografado com a chave de sessão TLS e um MAC é calculado sobre o texto criptografado.

O servidor armazena o TTA em seu banco de dados junto ao seu tempo de validade (i.e., um *time stamping*). Ele também armazena e relaciona o TTA com o IMEI do dispositivo, bem como com os outros dados do usuário. O servidor envia ao cliente o TTA criptografado e regido pelo MAC. Ao rebeber o TTA criptografado e o MAC, o cliente verifica o MAC e decriptografa o TTA. Em seguida, ele armazena o TTA para futuras autenticações com o servidor.

Uma vez realizado o primeiro acesso, o usuário pode conectar-se à Internet por meio do servidor. Para isso, após o estabelecimento de uma nova conexão TLS com o servidor, o cliente utiliza o mesmo algoritmo de criptografia simétrica anterior e a nova chave de sessão TLS para criptografar a concatenação dos seguintes dados: o TTA, o IMEI do dispositivo móvel, e o número do celular. O cliente então calcula o MAC sob a texto cifrado e os envia ao servidor. Ao receber esses dados, o servidor verifica o MAC e decifra o texto cifrado, obtendo, assim, o TTA, o IMEI, e o número do aparelho telefônico. O servidor, então, compara os dados recebidos com os armazenados. Caso os dados sejam os mesmos, o servidor autoriza o acesso à Internet pelo dispositivo.

Embora a proposta de [MELO and ARAUJO 2021] possibilite autorizar o acesso à Internet em redes públicas, ela requer refinamentos a fim de torná-la mais prática. Por exemplo, ao receber os códigos do servidor, o cliente precisa criptografá-los juntos a outros dados do usuário. No entanto, o protocolo não define como esses dados serão obtidos. Além disso, o canal TLS apresenta redundância de segurança, haja vista que os dados são criptografados e têm garantia de integridade via MAC. Assim, o TLS apenas fornece a chave de sessão para essas operações. Portanto, considerar a implementação do protocolo, incrementando melhorias, pode resultar em uma solução mais simples e viável.

4. Revisitando o Protocolo Para Liberação de Acesso Seguro a Redes Sem Fio Públicas Gratuitas

Considerando os pontos negativos enfatizados na seção 3, a fim de se aperfeiçoar a proposta de [MELO and ARAUJO 2021], a seguir é apresentada uma releitura de tal proposta. Tal estudo é realizado por meio da implementação do protocolo anterior, todavia considerando sua simplificação, novos recursos e aspectos práticos.

4.1. Requisitos, Modelo de Segurança e Limitações

A implementação do protocolo utiliza-se do esquema cliente e servidor. O cliente é um aplicativo desenvolvido para a plataforma Android por meio da linguagem Java. O servidor recebe, processa e retorna as requisições do cliente. O protocolo foi implementado na linguagem Python, por meio do *framework* Django [Django 2023] (*backend*). Dessa forma, o usuário utiliza esse aplicativo para obter e manter o seu token de acesso ao serviço. A comunicação entre o cliente e o servidor é realizada por meio de um canal seguro TLS. Parte da informação trocada entre o cliente e o servidor é codificada no formato JWT (ver Seção 2). O servidor também dispõe de um par de chaves pública e privada para assinatura digital.

A releitura introduzida possui aqui um modelo de segurança e limitações similares ao protocolo anterior, como descrito a seguir:

1. Entre os dados fornecidos pelo usuário, pelo menos o CPF foi devidamente validado durante o cadastro.
2. Existe um local apropriado e confiável para coleta dos dados dos usuários.
3. O identificador exclusivo da plataforma móvel não pode ser alterado. No caso do sistema Android, esse ID é o AndroidID.
4. Um atacante não deve ser capaz de monitorar canais externos como *e-mails* e *SMS*.
5. Um atacante não deve ser capaz de alterar a Identificação Internacional de Equipamento Móvel (IMEI).
6. Cada dispositivo possui um único cadastro.
7. A solução limita-se à geração de um token para acesso à Internet como associação entre o usuário e seu dispositivo. Ignora-se aqui etapas adicionais necessárias para se disponibilizar efetivamente a Internet no dispositivo.

4.2. A Implementação do Protocolo

Os detalhes da implementação do protocolo são apresentados a seguir. Ela considera dois cenários para obtenção dos dados do usuário. Em um deles, o usuário visita um local apropriado para esse fim. No outro, ele realiza o cadastro a partir de seu dispositivo. Além disso, a implementação diferencia o primeiro acesso do usuário ao serviço dos acessos posteriores.

Pré-Cadastramento de Usuários

A aplicação permite que os usuários sejam cadastrados previamente ou não. No primeiro caso, antes de realizar o primeiro acesso ao serviço, o usuário deve realizar o seu pré-cadastro. Para isso, ele visita pessoalmente um local apropriado para tal finalidade. Nesse

local, o usuário deve informar dados como o nome, o *e-mail*, o CPF e cadastrar uma senha para o primeiro acesso. Ele deve autorizar o armazenamento de seus dados conforme a LGPD (ver Seção 2.6). Os dados, portanto, são salvos no servidor da aplicação.

O cadastro presencial, no entanto, traz menos liberdade e comodidade aos usuários. Dessa forma, em um cenário mais prático, usuários realizam todo o processo de cadastro via rede. Nesse panorama, eles primeiramente efetuam o *download* do aplicativo; informam seus dados cadastrais diretamente nele, além da senha de primeiro acesso. Por exemplo, se forem requeridos nome e CPF, a aplicação deve verificar se o CPF é válido e se ele corresponde ao nome informado. O texto seguinte considera tal cenário como sendo o principal.

Primeiro Acesso

Em um primeiro acesso, o usuário não dispõe de uma conexão com a Internet e deseja obtê-la por meio do servidor do serviço. O usuário deve primeiramente realizar o *download* do aplicativo Android e instalá-lo em seu dispositivo móvel (e.g., *smartphone*).

Figura 2. Comunicação Cliente-Servidor do Protocolo.

Ao executar o aplicativo (ver Figura 3), ele primeiramente realiza uma conexão TLS com o servidor (Passo 1). O usuário então efetua o seu cadastro por meio da aplicação (Passo 2). Para isso, ele deve informar seu nome, seu *e-mail*, sua senha, seu CPF e sua data de nascimento. Além dos dados informados, o aplicativo também obtém o identificador *AndoidID* (ver Seção 2.4) do dispositivo móvel. Esses dados são enviados ao servidor.

Ao receber os dados do usuário, o servidor os armazena em um banco de dados. No entanto, a senha e o AndroidID são pré-processados antes do armazenamento. No caso da senha, o sevidor executa uma função de derivação de chave (e.g., Argon2 [Biryukov et al. 2015]) utilizando a senha informada pelo usuário e armazena a saída dessa função. Um processo semelhante ocorre com o AndroidID. Ele é utilizado como entrada para uma função criptográfica hash (e.g., SHA-256 [NSA 2002]) e a saída dessa função é armazenada no banco de dados.

Após armazenar esses dados, o servidor gera um token de acesso temporário. Esse token é composto por seis dígitos numéricos aleatórios e possui um tempo de validade limitado (e.g. 2 minutos). O servidor então envia o token gerado ao usuário. Considera-se aqui o *e-mail* como canal de envio desse token, entretanto outros canais podem ser utilizados, como o SMS.

Como etapa final de cadastro, o usuário deve informar o token de acesso temporário recebido. Esse token possibilita verificar se o usuário que realizou o cadastro é de fato o proprietário do *e-mail* fornecido. Caso o token corresponda ao mesmo enviado pelo servidor e esteja dentro de seu tempo de validade, o cadastro é confirmado e finalizado.

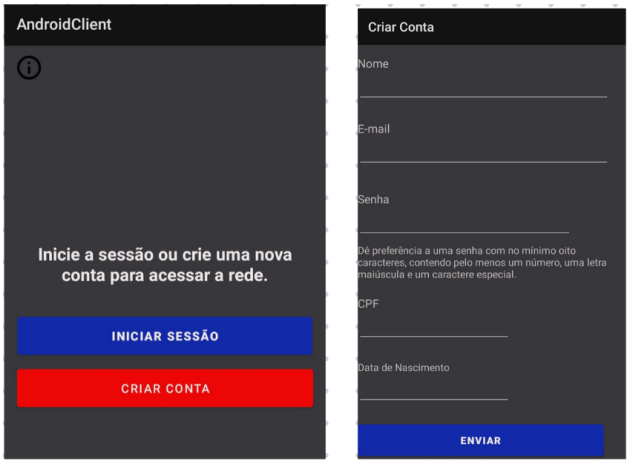

Figura 3. Aplicativo Android desenvolvido para acesso à redes WI-FI públicas.

Após a realização do cadastro, o usuário autentica-se no serviço para obtenção de seu token JWT. Esse será utilizado para acessar o serviço sem ser necessário autenticar-se novamente nos acessos posteriores. Para isso, o servidor gera um token JWT conforme descrito na Seção 2.5 e o assina com sua chave privada. A figura 4 apresenta o fluxo de cadastro do usuário em seu primeiro acesso à aplicação.

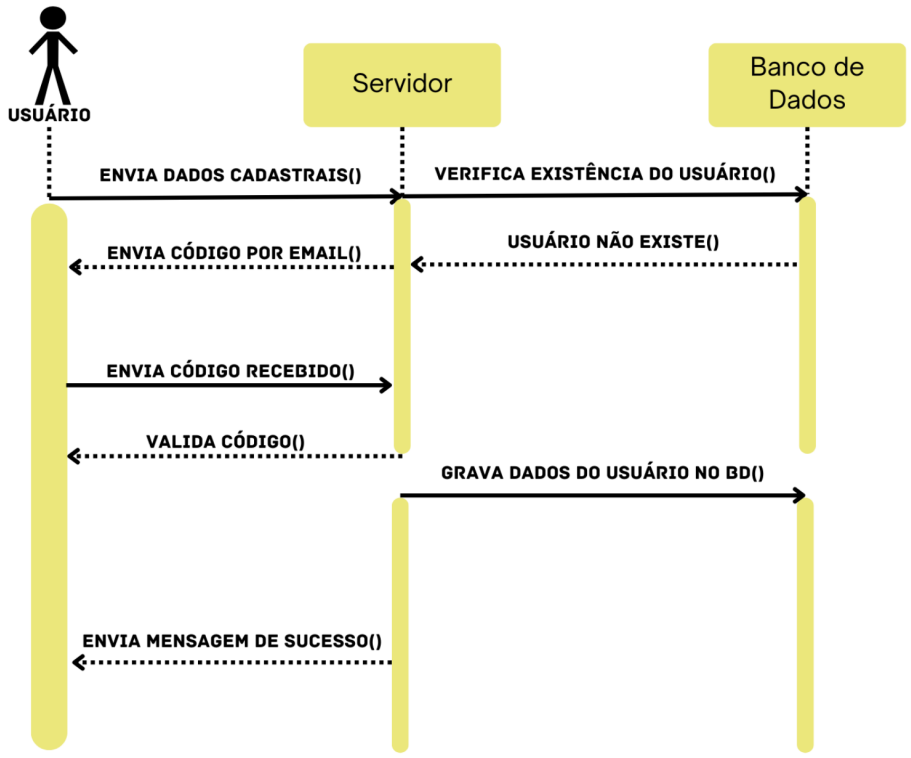

Figura 4. Diagrama de Sequência de cadastro do usuário.

Uma vez obtido o token de acesso JWT, o usuário recebe seu devido acesso aos recursos providos pela aplicação, i.e., o acesso à rede. Esse token JWT tem como *header* duas declarações. A primeira, o *alg*, define o algoritmo utilizado para assinar o token, ou seja, o algoritmo HS256 (HMAC com SHA-256). A segunda, o *typ*, define o tipo do

token. No *payload* do token, existem três declarações: o *id*, que define o ID do usuário; a *exp*, que define o tempo de validade do token JWT e a declaração *iat* que determina o *timestamp* da geração do token. O token JWT atribuído ao usuário é equivalente a um *cookie* de acesso de usuário [Barth 2011].

Acessos Posteriores

Quanto aos acessos posteriores do usuário ao aplicativo, enquanto ele possuir um token válido (i.e., dentro do período de validade e com assinatura válida), ele terá um *cookie* de acesso, cujo valor é o JWT. Diferentemente dos *cookies* de sessão clássicos, no caso do JWT, o servidor não armazena uma sessão no seu banco de dados ou em algum outro arquivo. Desde que a chave privada do servidor permaneça a mesma, o token continuará válido, a não ser que expire. Caso o usuário encerre a sua sessão antes do tempo de expiração do token, ele obterá um novo token em seu próximo acesso.

Visto que o JWT possui um tempo de validade equivalente a trinta dias nesse cenário, após tal período, a aplicação executará automaticamente o *logoff* do usuário, para que ele autentique-se novamente, e obtenha, assim, um novo token JWT. A figura 5 exemplifica a obtenção do token JWT por meio do *login* do usuário.

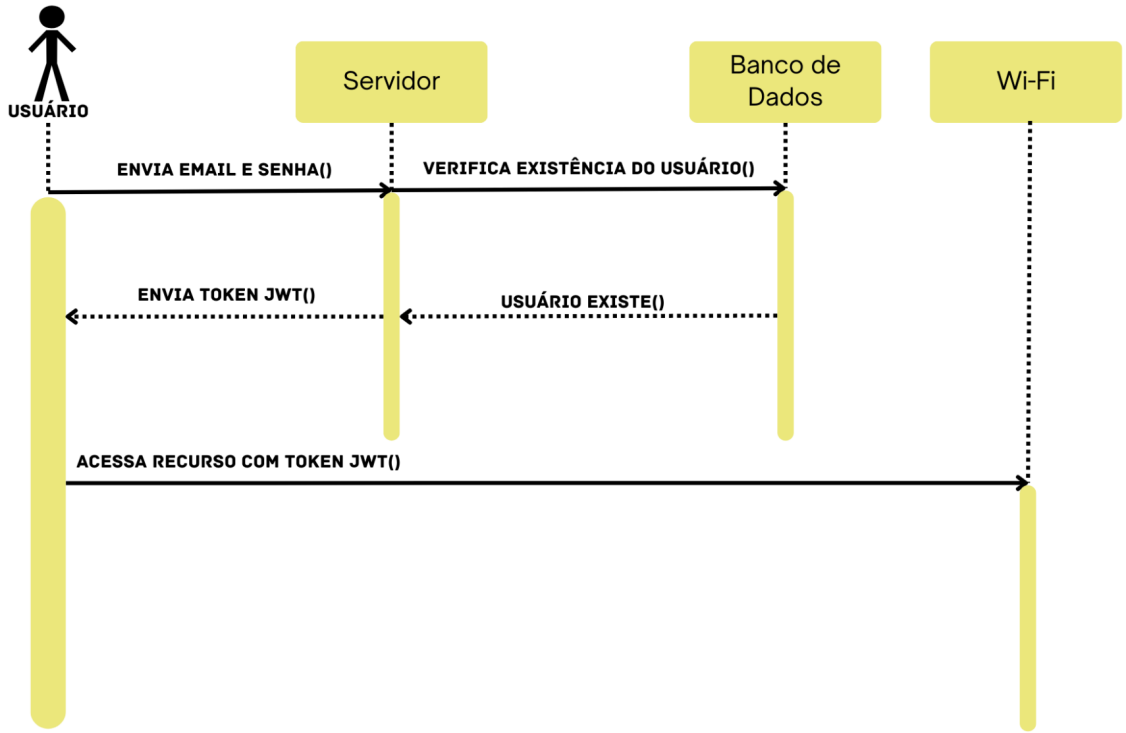

Figura 5. Diagrama de Sequência de autenticação do usuário.

4.3. Discussão

O protocolo proposto, uma vez implementado em redes sem fio públicas, visa possibilitar a identificação de atividades ilícitas, como incitação, produção ou posse de pornografia infantil, rouba e venda de dados de terceiros. Para isso, ele associa o usuário ao seu próprio *smartphone* tornando possível o rastreamento de ações suspeitas.

O objetivo do protocolo é atingido quando determinadas condições são satisfeitas. A primeira consiste na disposição do usuário em realizar o *download* e utilizar o aplicativo que permitirá o acesso à rede sem fio. A segunda consiste na correta validação das informações cadastrais do usuário, como o CPF, evitando que dados de outras pessoas sejam utilizados no ato de cadastro. A terceira condição diz respeito ao armazenamento dos dados de identificação do usuário, inclusive do identificador único do seu *smartphone*. Tal armazenamento é importante, pois agiliza o processo de identificação e de rastreio de potenciais transgressores da Lei.

Os dispositivos móveis dos usuários têm um papel fundamental para garantir a identificação de infratores. No entanto, em caso de furto de dispositivo móvel, não há como a aplicação distinguir entre o legitimo proprietário ou não, inviabilizando a correta identificação de um possível contraventor. Camadas adicionais de segurança, como autenticação biométrica, poderiam ajudar a identificação do legítimo usuário, mas estão fora do escopo desse trabalho.

Um outro aspecto importante é a substituição de dispositivos por seus usuários (e.g., a compra de um novo dispositivo). Como o sistema salva o identificador (AndroidID) do dispositivo, a substituição desse dispositivo resulta em um novo identificador. Nesse caso, a aplicação atual não dispõe de uma funcionalidade para substituir o dispositivo necessário. Assim, o usuário teria que realizar um novo cadastro.

Ainda em relação ao identificador AndroidID, a solução apresentada considera que esse valor não pode ser alterado. No entanto, isso poderia ocorrer durante a reinstalação do sistema Android, e.g., após a atualização da versão desse sistema. Uma solução simples para preservar o valor do AndroidID é a realização de um *backup* prévio. Por outro lado, ataques visando a alteração do AndroidID impediriam a associação entre o usuário e seu dispositivo. Como resultado, a solução não garantiria que o usuário que está utilizando a rede WI-FI é o mesmo que foi previamente cadastrado.

Com relação ao JWT, uma característica importante dessa tecnologia está no fato de que um token não pode ser invalidado, a menos que a chave privada do servidor (nesse caso, a que assinou o token) seja alterada ou que seu tempo de validade expire. Portanto, alterar a chave privada do servidor invalidaria todos os tokens gerados pela chave anterior, implicando em uma nova autenticação por parte dos usuários da rede. Por exemplo, se um usuário comete um determinado crime na rede, seu acesso não poderia ser imediatamente invalidado, pois não existe uma sessão desse usuário armazenada no banco de dados da aplicação.

Finalmente, é necessário que o usuário tenha acesso prévio à rede para realizar o seu cadastro. Isso poderia ser realizado de duas formas. A primeira é permitir um acesso mínimo à rede WI-FI apenas para que o usuário realize seu cadastro. Na segunda, o usuário deve dispor de acesso à Internet por meio de Internet móvel, por exemplo. Qualquer uma das formas poderia ser utilizada para a realização do cadastro inicial. Em caso de cadastro presencial não é necessária uma conexão prévia à rede.

5. Conclusão e Trabalhos Futuros

Tendo em vista que há cada vez mais estabelecimentos públicos e comerciais disponibilizando acesso à Internet por meio de redes WI-FI, a possível utilização dessas redes para

fins ilícitos requer a utilização de meios para identificar seus autores. Visando identificar os autores de atividades ilícitas nessas redes, este trabalho apresentou a implementação de um protocolo para esse fim. Tal implementação revisitou o protocolo anterior considerando aspectos práticos.

A solução introduzida aqui associa usuários a seus dispositivos móveis. Para isso, ela requer o cadastramento prévio dos usuários e o armazenamento desses dados. Embora a solução proposta possibilite identificar usuários em caso de atividades ilícitas, ela não é capaz de resistir a ataques em que usuários utilizam dados de terceiros para realizar o cadastro remoto ou ataques em que o identificador do dispositivo é substituído.

A resistência a tais ataques é fundamental para garantir a correta identificação de usuários nessas redes. A proposta de técnicas para resistir a tais ataques é deixada como trabalho futuro. Ademais, testes da aplicação em ambientes reais seriam também muito úteis. Mais além, convidar usuários voluntários para que utilizem a aplicação seria uma ótima forma de verificar pontos a serem aperfeiçoados, do ponto de vista do usuário. Tais testes qualificam, assim, trabalhos futuros.

Referências

Android, G. (2023). Visão geral do kernel. `https://source.android.com/devices/architecture/kernel?hl=pt-br`. Acessado em 25 de maio de 2023.

Barth, A. (2011). HTTP State Management Mechanism. RFC 6265.

Biryukov, A., Dinu, D., and Khovratovich, D. (2015). Argon2: the memory-hard function for password hashing and other applications. *IEEE European Symposium on Security and Privacy (EuroSP)*.

Certisign (2023). O que é Certificado Digital. `https://www.certisign.com.br/certificado-digital`. Acessado em 10 de março de 2023.

Developers, G. (2023a). Alterações de privacidade no android 10. `https://developer.android.com/about/versions/10/privacy/changes?hl=pt-br`. Acessado em 06 de março de 2023.

Developers, G. (2023b). ANDROID_ID. `https://developer.android.com/reference/android/provider/Settings.Secure#ANDROID_ID`. Acessado em 05 de março de 2023.

Developers, G. (2023c). Práticas recomendadas para identificadores exclusivos. `https://developer.android.com/training/articles/user-data-ids?hl=pt-br`. Acessado em 20 de março de 2023.

Dirks, T. and Allen, C. (1999). The TLS Protocol Version 1.0. RFC 2246.

Django (2023). Django makes it easier to build better web apps more quickly and with less code. `https://www.djangoproject.com/`. Acessado em 20 de março de 2023.

Executivo, P. (2014). Lei nº 12.965, de 23 de abril de 2014. `https://www.planalto.gov.br/ccivil_03/_ato2011-2014/2014/lei/l12965.htm`. Acessado em 11 de março de 2023.

Executivo, P. (2018). LEI Nº 13.709, DE 14 DE AGOSTO DE 2018. https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm. Acessado em 11 de março de 2023.

Jones, M. B., Bradley, J., and Sakimura, N. (2015). JSON Web Token (JWT). RFC 7519.

Lucena, J. (2022). Wi-fi público e os crimes virtuais. https://www.drjonatas.com.br/wi-fi-publico-e-os-crimes-virtuais. Acessado em 20 de maio de 2023.

MELO, P. A. and ARAUJO, R. S. (2021). Uma Proposta de Protocolo Para Liberação de Acesso Seguro a Redes Sem Fio Públicas Gratuitas. TCC (Graduação em Ciência da Computação) - Faculdade de Computação, Universidade Federal do Pará. p. 11.

NSA (2002). SECURE HASH STANDARD. *Federal Information Processing Standards*.

Stallings, W. (2014). *Criptografia e Segurança de Redes*. Pearson, 6th edition.

Zimmer, K. (2022). Hotspot: conheça os perigos do Wi-Fi público e gratuito. https://www.lumiun.com/blog/hotspot-conheca-os-perigos-do-wi-fi-publico-e-gratuito/. Acessado em 14 de maio de 2023.