



Projeto de Graduação em Computação III

IDENTIFICAÇÃO CIVIL DIGITAL

Autor: Kauan Manzato do Nascimento

Orientador: Prof. Dr. Carlos Alberto Kamienski

Santo André – SP

Maio de 2022

ÍNDICE

RESUMO.....	3
1. INTRODUÇÃO.....	4
1.1 CONTEXTUALIZAÇÃO.....	4
1.2 PROBLEMAS.....	5
1.3 OBJETIVO GERAL.....	7
1.4 OBJETIVOS ESPECÍFICOS.....	7
2. REVISÃO BIBLIOGRÁFICA.....	8
2.1 IDENTIFICAÇÃO.....	8
2.2 CRIPTOGRAFIA DE CHAVE PÚBLICA.....	9
2.2.1 CRIPTOGRAFIA DE CURVAS ELÍPTICAS.....	10
2.3 ASSINATURAS DIGITAIS.....	12
2.4 CERTIFICADOS DIGITAIS.....	13
2.5 INFRAESTRUTURA DE CHAVE PÚBLICA (PKI).....	14
2.6 IDENTIDADE DIGITAL.....	15
3. SOLUÇÃO PROPOSTA.....	16
3.1 FERRAMENTAS.....	16
3.2 FUNCIONALIDADES.....	17
3.3 EMISSÃO DA CARTEIRA DE IDENTIDADE.....	17
3.3.1 CÓDIGO DA EMISSÃO DA CARTEIRA DE IDENTIDADE.....	18
3.3.2 DEMONSTRAÇÃO DO CÓDIGO.....	21
3.4 AUTENTICAÇÃO.....	24
3.4.1 AUTENTICAÇÃO DO CERTIFICADO DO CLIENTE TLS.....	25
3.5 CRIPTOGRAFIAR E DESCRIPTOGRAFIAR.....	26
3.6 ASSINAR E VALIDAR.....	26
4. TESTES.....	28
4.1 ELEIÇÕES NO BRASIL.....	28
4.2 APLICAÇÃO DO ELEITOR.....	30
4.3 APLICAÇÃO DO TSE.....	37
4.4 DEMONSTRAÇÃO DAS ELEIÇÕES DIGITAIS.....	43
5. CONCLUSÃO.....	47
6. REFERÊNCIAS BIBLIOGRÁFICAS.....	48

RESUMO

Este trabalho discorre do problema de identificação civil no Brasil, expondo as suas causas e consequências e propondo uma solução baseada em criptografia de chave pública e na solução implementada na Estônia, porém adaptando à realidade brasileira. Depois de uma extensa revisão do problema, da bibliografia e de soluções semelhantes implementadas em outros países, há também uma demonstração da solução, com código disponível em um repositório do GitHub, e aplicação da solução em um cenário hipotético de eleições digitais.

O objetivo final deste trabalho, além de propor uma solução a um problema real, é complementar e estender a formação do aluno e avaliar o desempenho do discente tendo em vista os objetivos gerais do curso, conforme a disposição do PGC em 2022. Portanto, como será explicado posteriormente, este trabalho não resolve o problema e nem garante que a solução é a mais segura, apenas demonstra que é possível utilizá-la na prática para resolver o problema. Por fim, com os testes, é demonstrado que a solução é viável, simples e transparente. Porém, ainda faltarão algumas informações, como o impacto social e político, que este trabalho não trata por estar fora do escopo.

1 – INTRODUÇÃO

Este trabalho trata sobre o problema da identificação civil no Brasil e suas consequências e propõe como solução um sistema digital que faz a identificação e a autenticação dos cidadãos brasileiros baseado em infraestrutura de chave pública (PKI) e no sistema atual de identificação digital da Estônia, sendo avaliado pelos seguintes critérios:

- **Eficácia:** se o sistema cumpre os objetivos designados;
- **Eficiência:** quantidade de recursos utilizados para implementar o sistema, mantê-lo e utilizá-lo;
- **Interoperabilidade:** a solução precisa definir um padrão com abrangência nacional e que permita a interação entre os diferentes órgãos públicos;
- **Segurança:** a solução deve garantir a segurança (confidencialidade, integridade e autenticidade) das informações processadas e transmitidas;
- **Limitações:** as limitações e desvantagens conhecidas da solução proposta, em comparação com os benefícios trazidos por ela.

Os impactos políticos e sociais, legislação e regulamentações estão fora do escopo deste trabalho, ficando limitadas a trabalhos futuros. Além disso, uma possível aplicação da solução em um cenário hipotético de eleições digitais também é demonstrada, como forma de exemplificar e dissecar os detalhes da proposta.

1.1 – CONTEXTUALIZAÇÃO

Com o advento da pandemia do novo coronavírus (Covid-19) em 2020, o Governo Federal brasileiro concedeu um benefício chamado Auxílio Emergencial cujo objetivo é minimizar o impacto da crise do vírus na população de baixa renda, de trabalhadores informais, de microempreendedores individuais e contribuintes individuais do INSS (Instituto Nacional do Seguro Social) [20]. Porém, para receber o benefício, a pessoa precisa cumprir uma série de requisitos, conforme a Lei nº 13.982/2020.

O cadastro dos dados e a liberação dos valores do auxílio à população se dá por meio de um software para smartphones, chamado Caixa Tem: o cidadão instala o aplicativo no dispositivo, informa seus dados pessoais (autodeclaração e identificação) e aguarda a aprovação

do Ministério da Cidadania [22]. Em caso de aprovação, o benefício é liberado para a pessoa na conta do aplicativo, que funciona como uma conta corrente.

Entretanto, desde o início do cadastro, as pessoas têm relatado que sujeitos receberam o benefício sem cumprir os requisitos ou que algum agente malicioso usou os dados de terceiros para receber o benefício indevidamente. Estas situações caracterizam o delito de estelionato, previsto no artigo 171 do Código Penal [4, 7].

1.2 – PROBLEMAS

Os problemas citados na seção anterior acontecem porque dados pessoais (CPF, nome, endereço, gênero, data de nascimento etc.) de muitos brasileiros podem ser encontrados na Internet, disponibilizados para venda e, geralmente, utilizados para fins ilícitos, como fraudes [55].

O dado pessoal que iremos tratar aqui é o Cadastro de Pessoas Físicas (CPF), registro instituído em 1965 por meio da Lei 4.862, de 29 de novembro de 1965, projetado exclusivamente para a verificação da contribuição do Imposto de Renda de Pessoa Física (IRPF). Atualmente, o CPF é mantido pela Receita Federal do Brasil. Entretanto, na prática, o CPF é usado para a identificação dos cidadãos e suas relações com órgãos públicos em alternativa ao Registro Geral (RG), a exemplo do projeto de Lei 1.422, de 2019. Em outras palavras, o CPF está sendo usado para fins diferentes daqueles previstos no projeto original. Isso acontece porque o Brasil não possui um sistema nacional para a identificação civil e o documento de identificação, o RG, possui problemas.

O Registro Geral (RG), ou carteira de identidade, é um documento utilizado para a identificação de pessoas físicas nascidas no Brasil e tem validade nacional [9]. Contudo, existem alguns problemas associados a este documento:

- Cada unidade federativa é responsável por emitir uma carteira de identidade diferente, sem nenhuma ligação entre si. E, como são 27 unidades federativas, cada cidadão brasileiro pode ter 27 RGs diferentes.
- Os dados que constam no documento variam de acordo com o órgão responsável pela emissão, ou seja, existe uma falta de padrão no documento ao longo do tempo.

Existem outros documentos que são usados para identificar uma pessoa física, como a Certidão de Nascimento, o Título de Eleitor, a Carteira de Habilitação (CNH) e o Registro Nacional de Estrangeiro (RNE) (usado por estrangeiros). Toda essa informação espalhada entre as instituições torna o processo de identificação ainda mais complexo e passível de vulnerabilidades. E, além dos documentos já existentes, surgiram propostas de criação de outro documento usado especificamente para a identificação como:

- **Registro Civil Único (RCU)** [11]
- **Registro de Identificação Civil (RIC):** surgiu em 1997 [12] e regulamentada 13 anos depois, em 2010 [14].
- **Registro Civil Nacional (RCN)** [15].
- **Identificação Civil Nacional (ICN)** [16].
- **Documento Nacional de Identificação (DNI)** [17, 51].
- **Biometria** como alternativa a documentos físicos [18].

Surgiram novas propostas por décadas e, mesmo assim, nenhuma realmente resolveu os problemas. Primeiro que as propostas demoram muito para saírem do papel, como é possível notar no projeto do RIC (Registro de Identificação Civil) que demorou 13 anos para ser regulamentado e ainda não temos mais informações sobre o projeto. Segundo que, conforme são criados documentos, adicionamos complexidade no sistema e ainda podemos inserir novas vulnerabilidades nele, que podem permitir ainda mais fraudes. Com isso, podemos enumerar os problemas do sistema de identificação civil no Brasil atualmente:

- **Falta de padrão:** não há documento e informações padronizadas para identificar civis. O que há é muitas informações diferentes dispersas em diferentes instituições públicas, como o Título de Eleitor, CNH, CPF etc.
- **Falta de centralização:** não há uma instituição pública responsável pela centralização dos processos e dos dados para a identificação de civis.
- **Problemas de projeto:** o documento destinado a identificar cidadãos (RG) possui falhas, como enumeramos acima.
- **Soluções improvisadas e temporárias:** por causa dos problemas na identificação, o CPF acaba sendo uma alternativa para identificar os cidadãos, dado esse que pode ser prejudicial nas mãos de criminosos. Há também muitos projetos para resolver esse problema, mas que nunca saem do papel.
- **Sistema complexo:** a falta de centralização nos dados e processos e de padrões torna a tarefa de identificar pessoas difícil e ineficiente.

- **Burocracia:** as propostas para solucionar problemas de interesse público demoram muito por conta da burocracia envolvida, como exemplificado pela proposta do RIC, que nunca saiu do papel desde 1995.

Todos esses problemas geram consequências sérias como fraudes, inconsistência de dados e vazamentos de dados pessoais [48] como foi exemplificado pelas fraudes envolvendo o Auxílio Emergencial.

1.3 – OBJETIVO GERAL

Este trabalho tem como objetivo geral projetar, desenvolver e implementar um sistema de informações que permita a identificação e a autenticação das pessoas de forma a evitar fraudes, vazamentos de dados e inconsistência dos dados, além de criar um cenário de eleição digital em que o sistema desenvolvido é utilizado, de forma a substituir o CPF dado como identificador e autenticador de cidadãos brasileiros.

1.4 – OBJETIVOS ESPECÍFICOS

O objetivo deste trabalho consiste em emitir carteiras digitais, implementar um mecanismo de identificação e autenticação com a carteira digital e demonstrar como ele funcionaria em um cenário hipotético de eleições digitais.

As carteiras digitais emitidas são compostas de uma chave privada protegida por uma senha de quatro dígitos (PIN), escolhido pelo proprietário da carteira neste caso, e seu respectivo certificado digital, dados pessoais (nome completo, CPF, sexo, foto, filiação e naturalidade), registro do local, data de emissão e data de validade da carteira digital. Por sua vez, o mecanismo de identificação e autenticação com a carteira digital consiste em certificado digital para a identificação e a assinatura digital para autenticação.

Com os objetivos acima concluídos, serão construídas duas aplicações (uma para o eleitor e outra para o TSE) para simular um cenário de eleições digitais, utilizando a carteira digital que foi criada.

2 – REVISÃO BIBLIOGRÁFICA

2.1 – IDENTIFICAÇÃO

A **identificação** é um processo essencial para a sociedade, porque garante a unicidade de um indivíduo e permite a prestação de contas (ou responsabilização), estabelecimento de confiança entre indivíduos e instituições (públicas ou privadas) e o estabelecimento de uma relação entre o indivíduo e as informações relacionadas a ele, e tudo isso sem ambiguidade.

O processo de identificação é usado em vários setores da sociedade. Por exemplo: a identificação de pessoas é usada por empresas para garantir que as informações passadas pelos consumidores sejam válidas, para que possam permitir as empresas de prover seus serviços e gerenciar eficientemente os negócios.

A identificação analisada neste trabalho é a **identificação de cidadãos** de uma nação ou território, cujo principal responsável são instituições públicas. A identificação de cidadãos em diversos países é feita através dos documentos de identidade, que podem valer para todo o território do país ou não, e podem ser compulsórios ou não compulsórios.

O objetivo da identificação civil é relacionar um indivíduo com informações associadas a ele. Os usos, formatos e políticas, entretanto, variam no espaço e no tempo. Por exemplo: no século XIX, a migração de pessoas na França era monitorada pela polícia através do uso de passaportes internos [21], e no ano de 1938, na Alemanha nazista, os judeus foram obrigados a usar um documento de identidade para fortalecer a opressão do governo sobre eles [3].

Além dos usos citados anteriormente, o **processo de identificação se tornou essencial no meio digital** por causa dos primeiros sistemas computacionais no começo do século XX. Um dos principais mecanismos de controle de acesso, o ACL (*Access-Control List*), por exemplo, foi implementado pela primeira vez em 1965 [53] como parte do sistema de arquivos Multics. Esse e outros mecanismos tiveram influência na identidade digital porque trouxe a estrutura popular nomes de usuários e senhas, que permite uma entidade gerenciar os acessos e permissões dos usuários do sistema.

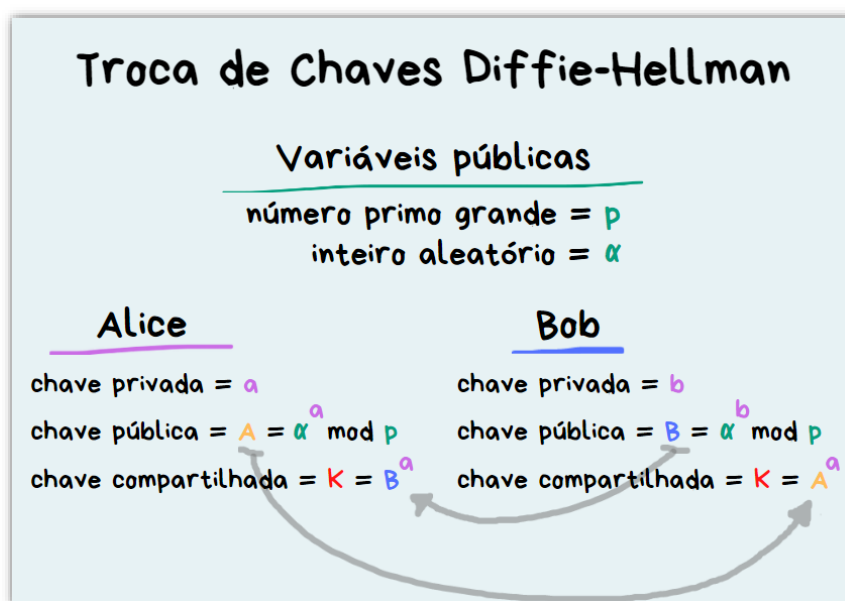
Outra forma de identificação em meios digitais surgiu na década de 1980, com o conjunto de protocolos TCP/IP. Os protocolos TCP/IP são a base da Internet atualmente e são responsáveis por atribuir endereços aos hosts da rede. Em outras palavras, o TCP/IP é uma forma de identificar os hosts da Internet.

2.2 – CRIPTOGRAFIA DE CHAVE PÚBLICA

É no período da criação dos protocolos TCP/IP (décadas de 1970 e 1980) que a criptografia se populariza como forma de garantir a confidencialidade, integridade e autenticidade das informações que passam pela Internet. A criptografia é o estudo e a prática de técnicas para a comunicação segura na presença de adversários [36]. Já o processo de criptografar, de acordo com o dicionário de Oxford, é o processo de converter dados em código, especialmente para prevenir o acesso não autorizado aos dados.

Na década de 1970, dois grupos independentes de cientistas inventaram o conceito de **criptografia de chave pública**. O primeiro grupo a inventar o conceito de criptografia de chave pública era formado pelos cientistas britânicos James H. Ellis, Clifford Cocks e Malcolm J. Williamson, membros do Quartel-General de Comunicações do Governo (GCHQ) britânico. Em 1970, Ellis idealizou a possibilidade de um sistema de criptografia não secreta, mas não viu um jeito de implementá-la. Em 1973, Cocks, colega de Ellis, criou um método prático para implementar o “sistema de criptografia não secreta” do amigo. Em 1974, Williamson criou um método de troca de chaves, que hoje conhecemos como troca de chaves Diffie-Hellman [25, 28] (Fig. 1). Entretanto, o trabalho do grupo ficou em segredo até 1997, quando o governo britânico liberou as descobertas [42].

Figura 1 – Troca de Chaves Diffie-Hellman



Fonte: Kauan Manzato do Nascimento, 2022.

A descoberta pública foi feita por cientistas norte-americanos, nos anos 1976 e 1977. Em 1976, Whitfield Diffie e Martin Hellman, influenciados pelo trabalho de Ralph Merkle sobre distribuição de chaves, inventaram um método de troca de chaves usando um conceito matemático chamado campo finito. Em 1977, Ron Rivest, Adi Shamir e Leonard Adleman inventaram o algoritmo RSA de criptografia de chave pública, publicado em 1978 [2] e que foi o estopim para a criação de muitos outros esquemas criptográficos semelhantes. No artigo publicado, também é descrito o conceito de assinatura digital, que usaremos mais para frente no trabalho.

De forma simplificada, a criptografia de chave pública funciona da seguinte forma: há dois tipos de chave, uma chave pública e uma chave privada (ou secreta). O emissor da mensagem usa a chave pública do destinatário para criptografar a mensagem, que apenas a chave privada pode descriptografar, sendo esta última em posse do destinatário. Em outras palavras, apenas o destinatário pode descriptografar a mensagem (Fig. 2). Assim, garantimos a confidencialidade da mensagem.

Figura 2 – Funcionamento da Criptografia de Chave Pública



Fonte: Kauan Manzato do Nascimento, 2022

2.2.1 – CRIPTOGRAFIA DE CURVAS ELÍPTICAS

Dentre os vários tipos de criptografia de chave pública, usaremos a criptografia de curvas elípticas, que é usado atualmente no sistema de identidade digital da Estônia [40]. Há

também outros tipos de criptografia de chave pública, como a criptografia baseada em números primos (RSA).

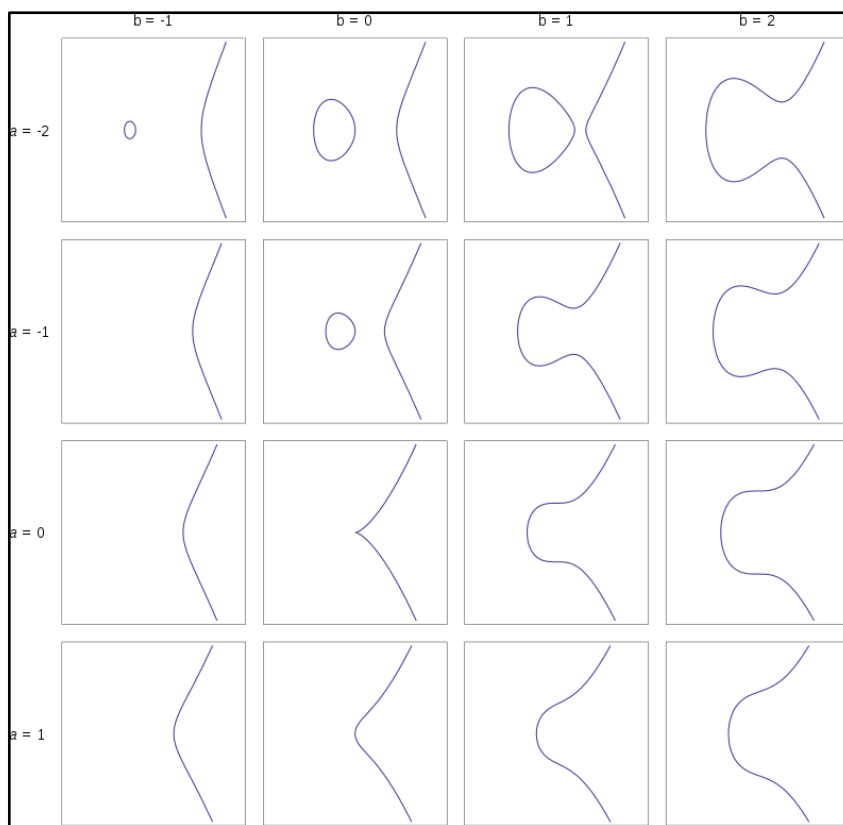
A criptografia de chave pública de curvas elípticas é baseado em um conceito matemático chamado de curva elíptica, por isso o nome. Em resumo, uma curva elíptica é um conjunto de pontos que satisfaz a seguinte equação:

$$y^2 = x^3 + ax + b, \quad a, b \in \mathbb{R}$$

Os parâmetros a, b e o campo finito usado definem diferentes curvas, com diferentes características e propriedades. Os pontos pertencentes às curvas e certas operações permitem criptografar e descriptografar informações.

Como a explicação técnica está fora do escopo, iremos nos limitar ao fato que já há curvas bem conhecidas e usadas, como descrito pela *Internet Assigned Numbers Authority* [34]. São alguns exemplos de curvas: secp384r1, brainpoolP512r1 e x25519. Neste trabalho, usaremos a curva NIST P-384 (secp384r1), definida no FISP 186-4, padrão usado pelo NIST, Instituto Nacional de Padrões e Tecnologia dos Estados Unidos, que também é a curva usada pela Estônia [24].

Figura 3 – Exemplos de curvas elípticas com diferentes parâmetros



Fonte: Wikimedia Commons, 2008.

2.3 ASSINATURAS DIGITAIS

Para garantir a autenticação da identidade do emissor, a criptografia de chave pública permite a criação de **assinaturas digitais** através de chaves secretas. Como as chaves são secretas, elas não podem ser forjadas e quem assinou não pode negar a assinatura [2]. Portanto, as assinaturas garantem identificação e autenticação e, por isso, são parte essencial da solução proposta neste trabalho, como veremos mais para frente.

Criar uma assinatura digital é semelhante à operação de criptografia, porém há diferenças. Em vez de usar a chave pública do destinatário, o remetente usa sua própria chave privada para criar a assinatura. Ao enviar a mensagem, o remetente envia também o certificado digital (com a chave pública) e a assinatura, assim o destinatário consegue verificar a autenticidade e a integridade da mensagem e identificar o remetente.

Em síntese, a assinatura consiste em calcular o hash da mensagem e criptografar o resultado com a chave privada do remetente. Assim, a assinatura pode ser verificada com a chave pública do remetente. Este mecanismo é mostrado pela Figura 4.

Figura 4 – Assinaturas digitais

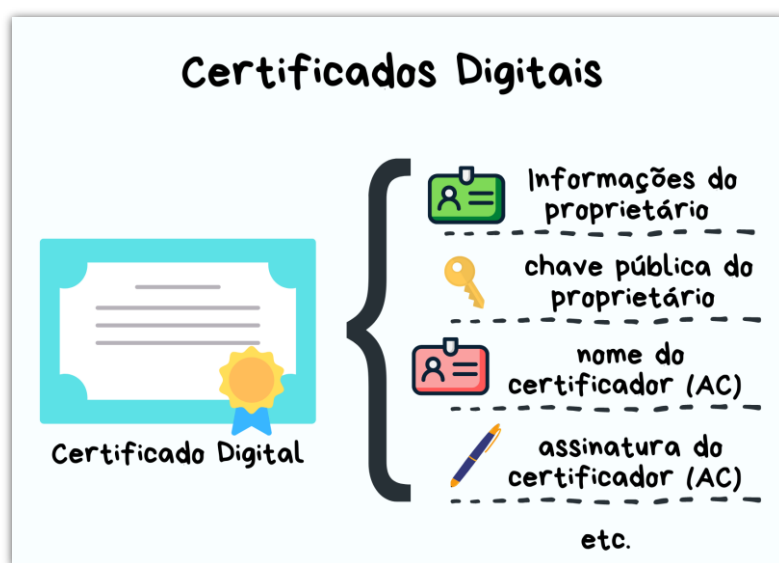


Fonte: Kauan Manzato do Nascimento, 2022

2.4 – CERTIFICADOS DIGITAIS

Em 1988, surge o **certificado digital** com a publicação da primeira versão do padrão X.509 [33]. Os certificados digitais são documentos eletrônicos usados para provar a identidade do proprietário de uma chave pública, e incluem as informações do proprietário, sua assinatura digital e sua chave pública (Fig. 5). Assim, o certificado digital auxilia a implementação da criptografia de chave pública [32], como veremos a seguir.

Figura 5 – Certificados digitais



Fonte: Kauan Manzato do Nascimento, 2022

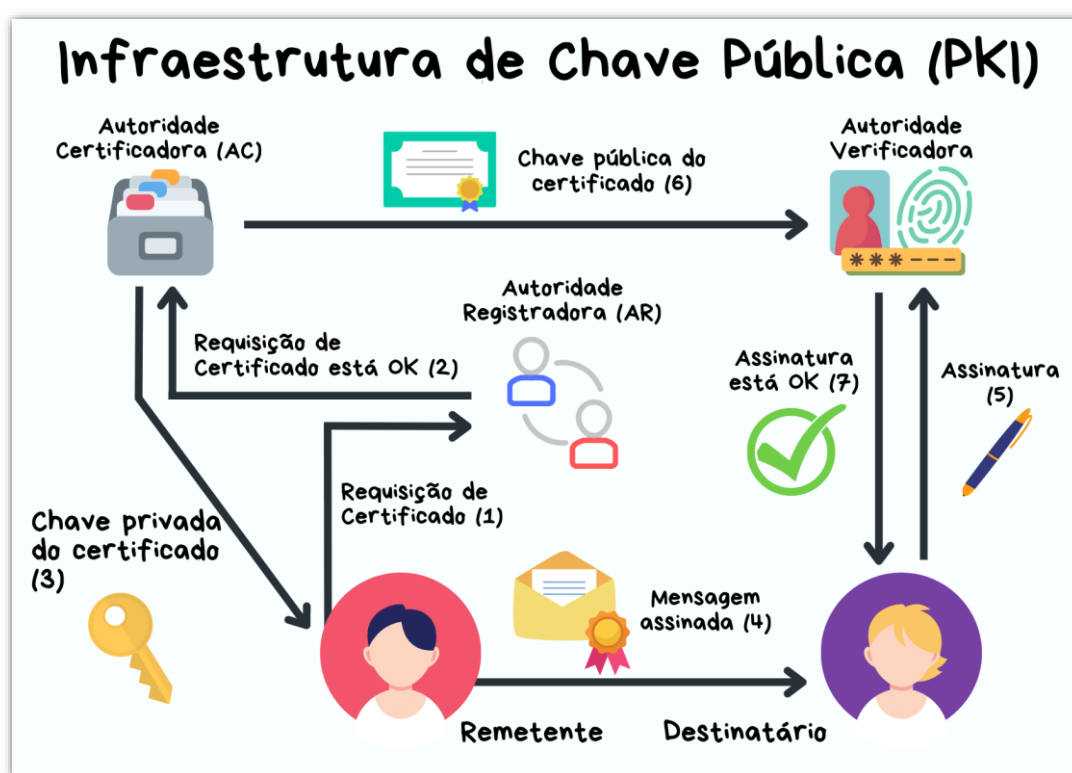
Na década de 1990, a popularização da Internet trouxe a necessidade de comunicações mais seguras, e foi assim que surgiu o protocolo SSL (Secure Sockets Layer). O SSL é um protocolo criptográfico publicado em 1995 na sua versão 2.0 [52], desenvolvido para garantir segurança das comunicações sobre redes de computadores, atuando na camada de aplicação e sendo usado principalmente no protocolo HTTPS, amplamente usado até os dias atuais na Internet.

Atualmente, o sucessor do SSL é o TLS, com sua versão mais recente sendo o TLS 1.3, publicado em 2018 [46]. O protocolo TLS/SSL usa certificados digitais e criptografia de chave pública: cliente e servidor trocam certificados digitais que permitem a confiança mútua entre cliente e servidor. Essa troca de certificados e chaves e autenticação é conhecido como *TLS handshake* e é o que garante a segurança do protocolo TLS, sendo uma aplicação popular de certificados digitais.

2.5 – INFRAESTRUTURA DE CHAVE PÚBLICA (PKI)

Ainda na década de 1990, com o crescimento do interesse por comunicações seguras pela Internet, surgiu o conceito de **infraestrutura de chave pública** (ou PKI) [56]. Uma infraestrutura de chave pública é um termo usado para se referir a um conjunto de hardware, software, políticas, procedimentos e processos usados para gerenciar certificados digitais e chaves para serem usados no esquema de criptografia de chave pública (Fig. 6).

Figura 6 – Funcionamento básico de uma Infraestrutura de Chave Pública (PKI)



Fonte: Kauan Manzato do Nascimento, 2022

Em 2001, foi instituído o órgão público no Brasil responsável pela certificação digital no país, chamado de **ICP-Brasil** (Infraestrutura de Chaves Públicas Brasileira [13]. O ICP-Brasil é uma estrutura hierárquica composta de várias Autoridades Certificadoras (AC), essas que são as entidades responsáveis por gerenciar e emitir certificados digitais. Uma dessas Autoridades Certificadora é a AC-Raiz (papel realizado pelo Instituto Nacional de Tecnologia da Informação), que credencia e audita as ACs do ICP-Brasil.

2.6 – IDENTIDADE DIGITAL

Em 2002, o governo da Estônia introduziu seu primeiro **documento de identidade digital** que usa criptografia de chave pública e certificados digitais para que os cidadãos estonianos se identifiquem e assinem digitalmente. O documento possui validade jurídica, assim como as assinaturas digitais criadas a partir das chaves do ID-card, como é chamado o documento. E é com base no projeto estoniano e seus resultados que este trabalho é desenvolvido. O documento é considerado um sucesso, já que 99% dos estonianos o possuem [30] e o documento é usado em várias atividades cotidianas dos cidadãos, como internet banking, assistência médica e até eleições (Fig. 7).

Figura 7 – carteira de identidade estoniana de 2021



Fonte: Estonian Police and Border Guard Board

Durante esses quase 20 anos de ID-card, o formato do documento e o chip dentro dele mudaram várias vezes. Entretanto, o funcionamento continuou basicamente o mesmo [40]: o ID-card tem duas chaves privadas com seus respectivos certificados digitais X.509, e chaves simétricas para operações usadas pela fabricante.

Além da Estônia, há outros países que também usam a identificação digital como, por exemplo:

- Bélgica [6]
- Cazaquistão [23]
- Holanda [31]
- Itália [35]
- Espanha [27] etc.

3 – SOLUÇÃO PROPOSTA

Conforme escrito anteriormente, a solução proposta é um cartão inteligente baseado no modelo estoniano, adaptando algumas características ao contexto brasileiro, e que serve como uma fonte de **autenticação** (chave de autenticação e certificado digital), **identificação** (informações pessoais gravadas digital e fisicamente. e certificado digital) e **confidencialidade** (funções criptográficas), a fim de criar uma alternativa às soluções atuais no Brasil.

3.1 – FERRAMENTAS

Para o desenvolvimento da solução deste trabalho, usamos a biblioteca de funções criptográficas **cryptography** para Python, que inclui interface a diversos algoritmos, cifras simétricas e assimétricas, funções para gerar chaves e *message digests*, como o SHA-3.

A biblioteca **cryptography**, por sua vez, é baseada na biblioteca OpenSSL. O OpenSSL é uma biblioteca de código aberto¹ escrita em C e que implementa duas bibliotecas: biblioteca de criptografia, que provê funções criptográficas como AES, RSA, SHA-3 etc., e a biblioteca TLS/SSL que implementa o protocolo TLS (SSL, TLS 1.2, TLS 1.3, DTLS etc.). O OpenSSL possui algumas alternativas, como o LibreSSL, BoringSSL e Google Tink, todos de código aberto, baseados no OpenSSL e que têm como objetivo resolver alguns problemas do OpenSSL, mas que são menos populares.

O Python está sendo usado porque permite prototipagem rápida, por ser uma linguagem de programação interpretada e ter uma sintaxe com um nível mais alto que outras linguagens, como por exemplo C ou Java.

E, por fim, usamos o IDLE (*Integrated Development and Learning Environment*), um ambiente de desenvolvimento integrado ao Python que possui interface gráfica e que permite a execução e a criação do código. O IDLE vem junto com o Python, por padrão, e está disponível em diferentes sistemas operacionais, como o Windows, o Linux e o MacOS.

¹ O código-fonte do OpenSSL está disponível em <https://github.com/openssl/openssl>.

3.2 – FUNCIONALIDADES

As funcionalidades inclusas na carteira de identidade digital são duas chaves assimétricas (ECC) com seus respectivos certificados X.509 de chave-pública correspondentes, além da chave simétrica para a realização de operações de manutenção no cartão e informações pessoais.

Chave de autenticação: uma das chaves privadas é a chave de autenticação. Essa chave é usada para se autenticar em serviços on-line ao providenciar uma assinatura digital no processo de autenticação do certificado TLS do cliente. A chave também permite descriptografar arquivos que foram criptografados para o proprietário do cartão, o que não é muito usado, já que estes arquivos ficariam ilegíveis caso o cartão fosse perdido ou destruído.

Chave de assinatura digital: a outra chave privada é a chave de assinatura digital. Essa chave é usada para vincular assinaturas digitais que, no modelo estoniano e sobre a regulamentação europeia eIDAS, são reconhecidas como assinaturas válidas.

Chave simétrica: as chaves de criptografia simétrica são pré-carregadas no cartão para que o fabricante possa realizar algumas operações depois da emissão do cartão, como resetar os códigos PIN, gerar novas chaves e escrever novos certificados.

Informações pessoais: no modelo estoniano, o chip do cartão também contém exatamente as mesmas informações que estão impressas no cartão, incluindo o número de identificação pessoal, chamado PIC, equivalente ao nosso número do RG ou CPF [29].

3.3 – EMISSÃO DA CARTEIRA DE IDENTIDADE

A carteira de identidade, portanto, seria um cartão inteligente com informações pessoais gravadas fisicamente no cartão, como nome e data de nascimento, e digitalmente no chip, além dos pares de chave e certificado. Por ora, para fins de demonstração, iremos nos limitar a simular a carteira apenas emitindo as informações digitais da carteira e um par de chave privada e certificado digital.

Com relação às informações pessoais, como elas estão visíveis fisicamente no cartão, elas podem ser gravadas em claro (sem criptografia) no cartão para a leitura fácil. Além disso, como essas informações são basicamente texto, o tamanho total dos arquivos não passa de 1 kB.

As chaves privadas armazenadas são protegidas, cada uma protegida por um PIN de 4 dígitos diferente. Uma das chaves, conforme descrito, é usada para autenticação e a outra, para assinatura. Porém, como este trabalho tem a finalidade de demonstrar a solução, iremos emitir apenas uma chave, para manter a simplicidade do projeto e melhorar a didática. A chave privada é uma chave privada de criptografia de curva elíptica (NIST P-384) de 384 bits e criptografada. Seu tamanho é de apenas 379 bytes. Assim, as duas chaves terão um tamanho de 758 bytes. Elas são geradas usando a biblioteca `cryptography` para Python e são salvas no formato PEM (*Privacy Enhanced Mail*).

Por fim, os certificados digitais estão no padrão X.509 e são criados a partir do par de chaves pública e privada. Ele contém as informações do emissor, do proprietário e da chave pública do proprietário, data de validade, número de série e outras informações.

Em aplicações reais, os certificados digitais são emitidos por entidades chamadas Autoridades Certificadoras (ACs) e possuem uma cadeia de confiança para serem válidos. Por exemplo: o certificado de um brasileiro é emitido por um cartório que, por sua vez, possui um certificado digital emitido pelo ICP-Brasil. Essa cadeia permite o seguinte: se houver confiança no ICP-Brasil, então é possível confiar no cartório e, por consequência, confiar no certificado digital da pessoa. Para simplificar o projeto, porém, iremos pular a cadeia de confiança e gerar certificados auto assinados.

3.3.1 – CÓDIGO DA EMISSÃO DA CARTEIRA DE IDENTIDADE

A emissão da carteira digital, neste trabalho, consiste um pequeno programa escrito em Python que recebe algumas informações e emite os dados da carteira, incluindo principalmente a chave privada e o certificado digital, cujo código é analisado a seguir. Em uma aplicação real, este programa poderia ser executado em um cartório, por exemplo, uma instituição oficial que poderia emitir a carteira de identidade.

Primeiramente, o usuário é recebido com o cabeçalho (função `cabecalho`) (Fig. 8) e com um pedido para inserir o nome do proprietário (função `set_owner_name`) (Fig. 9) e, depois, o nome da autoridade certificadora que vai emitir a carteira de identidade (função `set_issuer_name`) (Fig. 10).

Figura 8 – Definição da função `cabecalho`

```
def cabecalho():
    print('*****')
    print('*')
    print('*   EMISSÃO DA CARTEIRA DE IDENTIDADE DIGITAL   *')
    print('*')
    print('*****')
```

Fonte: Kauan Manzato do Nascimento, 2022.

Figura 9 – Definição da função `set_owner_name`

```
def set_owner_name():
    name = str(input("\nQual é o seu nome?\n"))
    return name
```

Fonte: Kauan Manzato do Nascimento, 2022.

Figura 10 – Definição da função `set_issuer_name`

```
def set_issuer_name():
    name = str(input("\nIdentificação do emissor da carteira de identidade:\n"))
    return name
```

Fonte: Kauan Manzato do Nascimento, 2022.

Segundo, é definido o PIN, que é escolhido pelo proprietário da carteira neste caso, e que protegerá a chave privada, utilizando a função `set_pin` (Fig. 11). Nesta função, é usado o conceito de expressão regular para validar a entrada do usuário.

Figura 11 – Definição da função `set_pin`

```
def set_pin():
    while(True):
        pin = input('\nEscreva um PIN de 4 dígitos para proteger sua chave privada:\n')
        if re.match(r"^\d{4}$", pin):
            return pin.encode()
        else:
            print('\nPIN inválido.')
```

Fonte: Kauan Manzato do Nascimento, 2022.

Então, geramos a chave privada com a função `gen_key` que usa a função `generate_private_key`, disponibilizada pela biblioteca `cryptography`, para gerar uma chave de criptografia de curvas elípticas usando a curva NIST P-384 (Fig. 12).

Figura 12 – Definição da função `gen_key`

```
def gen_key():  
    return ec.generate_private_key (  
        ec.SECP384R1()  
    )
```

Fonte: Kauan Manzato do Nascimento, 2022.

Depois, a partir da chave privada, extraímos a chave pública usando o método `public_key` das chaves privadas (Fig. 13).

Figura 13 – Extração da chave pública

```
public_key = private_key.public_key()
```

Fonte: Kauan Manzato do Nascimento, 2022.

No próximo passo, geramos o certificado digital com a função `gen_cert`, passando como argumentos o nome do proprietário, nome do emissor e a chave privada. Esta função define os parâmetros do certificado como o nome do proprietário, o nome do emissor, a chave pública, número serial etc. (Fig. 14).

Figura 14 – Definição da função `gen_cert`

```
def gen_cert(owner_name, issuer_name, private_key):  
    name_attribute_owner = x509.Name([x509.NameAttribute(NameOID.COMMON_NAME, owner_name)])  
    name_attribute_issuer = x509.Name([x509.NameAttribute(NameOID.COMMON_NAME, issuer_name)])  
    now = datetime.utcnow()  
  
    cert = (  
        x509.CertificateBuilder()  
        .subject_name(name_attribute_owner)  
        .issuer_name(name_attribute_issuer)  
        .public_key(private_key.public_key())  
        .serial_number(1000)  
        .not_valid_before(now)  
        .not_valid_after (now+timedelta(days=10*365))  
        .add_extension(x509.BasicConstraints(ca=True, path_length=0), False)  
        .sign(private_key, hashes.SHA384(), default_backend())  
    )  
  
    return cert
```

Fonte: Kauan Manzato do Nascimento, 2022.

Depois, codificamos a chave privada e o certificado no formato PEM, além de criptografar a chave privada com o PIN escolhido (Fig. 15).

Figura 15 – Conversão da chave privada e do certificado para o formato PEM

```
cert_pem = cert.public_bytes(encoding=serialization.Encoding.PEM)
private_key_pem = private_key.private_bytes(
    encoding = serialization.Encoding.PEM,
    format = serialization.PrivateFormat.TraditionalOpenSSL,
    encryption_algorithm = serialization.BestAvailableEncryption(pin)
)
```

Fonte: Kauan Manzato do Nascimento, 2022.

Então salvamos o certificado e a chave privada junto com as informações pessoais (neste caso, usamos o nome do proprietário para exemplificar as informações pessoais, mas podem ser incluídas outras informações) com a função `save_file` (Fig. 16), finalizando assim a simulação da emissão da carteira de identidade.

Figura 16 – Definição da função `save_file` e seu uso no código

```
def save_file(path, data):
    with open(path, 'wb') as file:
        file.write(data)
```

```
save_file('files/certificado.pem', cert_pem)
save_file('files/chave.pem', private_key_pem)
save_file('files/nome', bytes(owner_name, encoding='utf-8'))
```

Fonte: Kauan Manzato do Nascimento, 2022.

Analisando o código, é possível ver muitos pontos de melhoria, obviamente. Mas, para os pontos que o trabalho se propôs a demonstrar, o código é suficiente, mostrando que a solução é simples, transparente e segura. Além disso, o código está disponível de forma atualizada no GitHub².

3.3.2 DEMONSTRAÇÃO DO CÓDIGO

O código escrito para a emissão da carteira digital pode ser executado com qualquer interpretador de Python e consiste em um programa sem interface gráfica que, apesar de deixar o programa mais intuitivo também tornaria o projeto mais complexo sem contribuir com a proposta do trabalho. Neste caso, vamos usar o terminal do Windows para executar o código (Fig. 17).

² O código-fonte deste trabalho está disponível em: <https://github.com/kauanmn/PGC>.

Figura 17 – Execução do programa de emissão da carteira de identidade

```
PS D:\UFABC\PGC FINAL\Código\carteira digital> python .\emissao.py
*****
*                                                                    *
*  EMISSÃO DA CARTEIRA DE IDENTIDADE DIGITAL  *
*                                                                    *
*****

Qual é o seu nome?
Kauan Manzato do Nascimento

Identificação do emissor da carteira de identidade:
Cartorio de Santo Andre

Escreva um PIN de 4 dígitos para proteger sua chave privada:
3011

Gerando chave privada...
Chave gerada com sucesso!
Gerando certificado...
Certificado gerado com sucesso!
Salvando informações usando a codificação PEM...
Arquivos salvos com sucesso.
```

Fonte: Kauan Manzato do Nascimento, 2022.

O programa recebe como entrada as informações pessoais, no caso o nome do proprietário da carteira de identidade, a identidade do emissor da carteira e o PIN que vai proteger a chave privada. Cada etapa do processo é impressa na tela até o fim (Fig. 18). Como resultado os arquivos são salvos na pasta `./files` (Fig. 19).

Figura 18 – Execução do programa de emissão da carteira de identidade




```
PS D:\UFABC\PGC FINAL\Código\carteira digital> dir .\files\

Diretório: D:\UFABC\PGC FINAL\Código\carteira digital\files

Mode                LastWriteTime         Length Name
----                -
-a----           27/04/2022   12:51           595 certificado.pem
-a----           27/04/2022   12:51           379 chave.pem
-a----           27/04/2022   12:51           27 nome
```

Fonte: Kauan Manzato do Nascimento, 2022.

Figura 19 – Arquivos criados pelo programa

Nome	Data de modificação	Tipo	Tamanho
 certificado.pem	27/04/2022 12:51	Privacy Enhanced Mail	1 KB
 chave.pem	27/04/2022 12:51	Privacy Enhanced Mail	1 KB
 nome	27/04/2022 12:51	Arquivo	1 KB

Fonte: Kauan Manzato do Nascimento, 2022.

Para mostrar o conteúdo do certificado e da chave privada de forma amigável, podemos usar a própria biblioteca OpenSSL (Fig. 21, 22). Já as informações pessoais são arquivos codificados em UTF-8 e podemos imprimi-la na tela usando `cat` (Fig. 20).

Figura 20 – Conteúdo do arquivo `nome`

```
PS D:\UFABC\PGC FINAL\Código\carteira digital\files> cat nome
Kauan Manzato do Nascimento
```

Fonte: Kauan Manzato do Nascimento, 2022.

Figura 21 – Conteúdo da chave privada `chave.pem`

```
PS D:\UFABC\PGC FINAL\Código\carteira digital\files> openssl ec -in .\chave.pem
read EC key
Enter PEM pass phrase:
writing EC key
-----BEGIN EC PRIVATE KEY-----
MIGkAgEBBDC6sEn0T0LWnhMVPjIbUgkjvVpL7zk60oZLAejbidzH3X2Pr8F+V7pj
xt86eb96NIigBwYFK4EEACKhZANiAAT5vcNbD507KYc4TVy1FbROInCmruecFQke
jDkSEpf+gu3Q4yq0tBFfJ2UxcY9Tp8uPCJIE0E98abNHwTsS906Slk/0hLGpUKy
vZX8VN6GK0thR6S3ZzJCPFPRDX6A7QI=
-----END EC PRIVATE KEY-----
```

Fonte: Kauan Manzato do Nascimento, 2022.

Figura 22 – Conteúdo do certificado digital

```
PS D:\UFABC\PGC FINAL\Código\carteira digital\files> openssl x509 -in .\certificado.pem -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1000 (0x3e8)
        Signature Algorithm: ecdsa-with-SHA384
        Issuer: CN = Cartorio de Santo Andre
        Validity
            Not Before: Apr 27 15:59:39 2022 GMT
            Not After : Apr 24 15:59:39 2032 GMT
        Subject: CN = Kauan Manzato do Nascimento
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
            Public-Key: (384 bit)
            pub:
                04:f9:bd:c3:5b:0f:9d:3b:29:87:38:4d:5c:b5:15:
                b4:4e:22:70:a6:ae:e7:9c:15:09:1e:8c:39:12:12:
                97:fe:82:ed:d0:e3:2a:b4:3a:d0:45:7c:9d:94:c5:
                c6:3d:4e:9f:2e:3c:22:48:78:e1:3d:f1:a6:cd:1f:
                04:ec:4b:dd:3a:4a:59:3f:d2:12:c6:a5:42:b2:bd:
                95:fc:54:de:86:28:eb:61:47:a4:b7:67:32:42:3c:
                53:d1:0d:7e:80:ed:02
            ASN1 OID: secp384r1
            NIST CURVE: P-384
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:TRUE, pathlen:0
        Signature Algorithm: ecdsa-with-SHA384
        30:66:02:31:00:81:a1:98:82:da:f9:2d:3d:33:be:2f:8c:a0:
        e3:b6:10:24:35:04:02:b0:98:78:95:19:20:79:8c:56:73:16:
        5f:11:0f:76:8a:3e:a8:a4:12:cd:0f:aa:cf:34:c2:3e:b0:02:
        31:00:cc:3f:6e:c2:7a:0b:43:f2:db:ea:69:a3:b0:2b:34:ac:
        c8:e0:02:18:dc:1d:33:1d:e5:d0:06:9a:9d:53:dd:93:f7:1a:
        84:41:a4:ea:0e:8c:5f:31:b6:7a:7d:a4:26:3b
-----BEGIN CERTIFICATE-----
MIIBiCCAQ6gAwIBAgICA+gwCgYIKoZIj0EAwwIjEgMB4GA1UEAwwXQ2FydG9y
aW8gZGUGU2FudG8gQW5kcmUwHhcNMjIwNDI3MTU1OTM5WWhcNMzIwNDI0
MTU1OTM5WjAmMSQwIgYDVQDDBtLlYXVhbiBhbnVW56YXRvIGRvIE5hc2NpbW
VudG8mdjAQBgcqhkjOPQIBBgUrgQQAIGNiAAT5vcNbD507KYc4TVy1FbR0In
CmruECFQkejdKSEpf+gu3Q4yq0tBFFJ2UxcY9Tp8uPCJIE0E98abNHwTsS906
SLk/hLGPuKyvZX8VN6GK0thR6S3ZJCPRDX6A7QKjEzARMA8GA1UdEwQIMAYBAf8CAQAwCgYIKoZIj0EA
AwMDaQAwZgIxAIGhmILa+S09M74vjkDjthAkNQCcsJh4LRkgeYxWcxZFEQ92Ij6o
pBLND6rPNMI+sAIXAMw/bsJ6C0Py2+ppo7ArNKzI4AIY3B0zHeXQBpqdU92T9xqE
QaTqDoxfMbZ6faQmOw==
-----END CERTIFICATE-----
```

Fonte: Kauan Manzato do Nascimento, 2022.

3.4 – AUTENTICAÇÃO

De acordo com o *National Institute of Standards and Technology* (NIST, Instituto Nacional de Padrões e Tecnologia dos Estados Unidos), a autenticação é o processo de verificar a identidade de um usuário, processo ou dispositivo, geralmente como pré-requisito para permitir o acesso a diferentes recursos dentro de um sistema de informação.

O caso de uso mais popular para a chave de autenticação da carteira de identidade estoniana, a qual a solução é baseada, é a autenticação em serviços Web pela Internet usando o protocolo TLS. Outros usos menos comuns para autenticação da identidade digital incluem assinatura de e-mails com S/MIME, autenticação SSH e VPN, e login em estações de trabalho [40]. O S/MIME (Secure/Multipurpose Internet Mail Extensions) é um padrão que permite criptografar e-mails [47]; a VPN (Virtual Private Network) é uma rede de computadores lógica

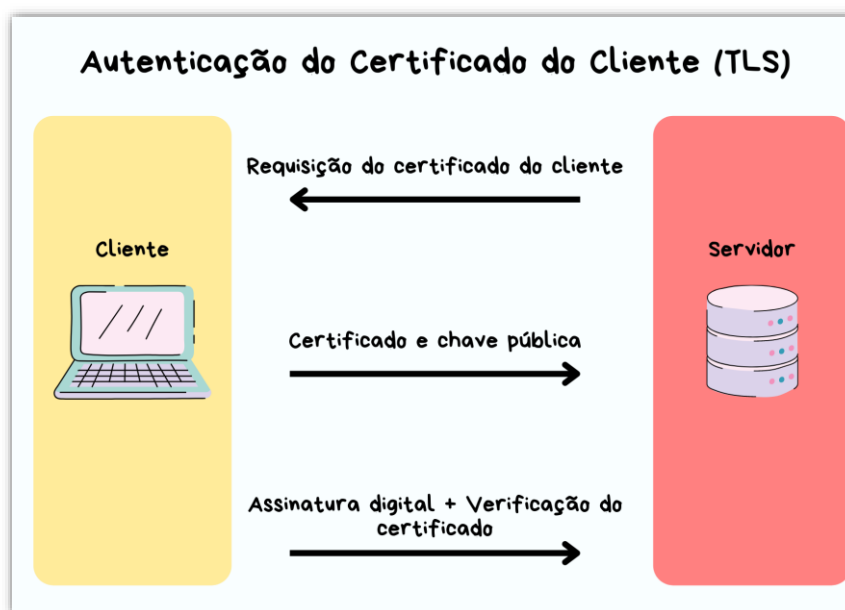
que usa recursos de uma rede física, geralmente usando criptografia e tunelamento [1]; e o SSH (Secure Shell) é um protocolo que permite operar serviços de uma rede insegura de uma forma segura [44].

Hoje, há planos em desenvolvimento para introduzir um novo método de autenticação para as carteiras de identidade digital através da autenticação a nível de aplicação usando uma extensão no navegador que assina um desafio com a chave de autenticação, de código aberto e disponível no GitHub³, chamado de Web eID.

3.4.1 – AUTENTICAÇÃO DO CERTIFICADO DO CLIENTE TLS

O protocolo TLS inclui autenticação do certificado do cliente (CCA), o que é suportado pela maioria dos navegadores e servidores TLS. Este método pode ser usado pelos provedores de serviços para implementar a autenticação dos usuários com o ID card e garante a autenticação do cliente e a integridade dos dados. Durante o processo CCA, o cliente envia seu certificado de autenticação para o servidor e prova sua identidade com uma assinatura digital, conforme definido pela [46] (Fig. 23).

Figura 23 – Autenticação do Certificado do Cliente (TLS)



Fonte: Kauan Manzato do Nascimento, 2022.

³ O código-fonte do Web eID está disponível em: <https://github.com/web-eid/web-eid-system-architecture-doc>.

3.5 – CRIPTOGRAFAR E DESCRIPTOGRAFAR

Um dos problemas da criptografia de curvas elípticas (ECC) é que não há formas práticas de criptografar e descriptografar dados como o RSA. Foram desenvolvidos três esquemas de ECC, mas cada uma com suas desvantagens, tornando-as impraticáveis. Por isso, a comunidade acadêmica abandonou esses esquemas e abraçou os sistemas híbridos de criptografia, onde curvas elípticas são usadas apenas para trocar chaves simétricas (e.g., troca de chaves com ECDH), sendo essas últimas as responsáveis por criptografar os dados [5].

Dentre os esquemas híbridos usados, o que está disponível em um maior número de padrões (ANSI X9.63, IEEE 1636rd, ISO/IEC 18033-2 e SECG SEC 1) é o ECIES (Elliptic Curve Integrated Encryption Scheme).

No caso da Estônia, a solução também é o sistema híbrido, onde chaves AES de 256 bits são derivadas de um segredo de 48 bytes compartilhado pelo ECDH (Elliptic Curve Diffie-Hellman). E, por fim, as informações são criptografadas com a cifra simétrica AES-256 GCM (Galois/Counter Mode) [40].

Neste trabalho em questão, serão usadas duas operações de criptografia durante o cenário das eleições, por causa do problema explicado acima:

- O voto será criptografado usando uma chave RSA de 4096 bits, previamente gerada. O voto é encriptado com a chave pública, que simula a chave pública do TSE, e é descriptografado com a chave privada e a respectiva senha.
- Os dados necessários que são enviados para o servidor do TSE (voto secreto, assinatura e certificado) serão criptografados usando o algoritmo ECDH (Elliptic Curve Diffie-Hellman). Este algoritmo permite a comunicação segura entre o usuário (eleitor) e o servidor (TSE).

3.6 – ASSINAR E VALIDAR

Como dito anteriormente, a assinatura digital prova a identidade do proprietário da carteira de identidade digital (autenticação), porque apenas o proprietário da chave privada consegue gerar uma assinatura digital válida. Neste trabalho, a assinatura será criada da seguinte forma: a mensagem é recebida em bytes, seu hash é calculado com SHA-384 e, por fim, a

assinatura é feita usando o algoritmo ECDSA (Elliptic Curve Digital Signature Algorithm) com a chave privada do usuário.

Para verificar a assinatura, o ECDSA realiza uma série de cálculos para poder validar a assinatura [37, 45]. Resumidamente, quem está validando a assinatura usa as informações presentes na chave pública para validar. A verificação bem-sucedida da assinatura significa a autenticidade e a integridade da mensagem.

4 – TESTES

Para demonstrar o uso da carteira digital, foi projetado um cenário de eleições digitais, onde a confidencialidade do voto deriva da criptografia de chave pública e a autenticidade é resultado da assinatura digital e a sua verificação. Os dados necessários do eleitor, como o certificado digital e a chave privada, foram emitidos previamente usando a aplicação de emissão da carteira de identidade.

4.1 – ELEIÇÕES NO BRASIL

As eleições no Brasil começam com o TSE. O TSE (Tribunal Superior Eleitoral) é a autoridade jurídica máxima da Justiça Eleitoral brasileira. As demais instâncias da Justiça Eleitoral é composta pelos juízes e juntas eleitorais e pelos TREs (Tribunal Regional Eleitoral). Junto com os TREs, **o TSE é responsável pela gestão das eleições no Brasil** [10].

As eleições para Presidente da República, governadores, senadores e prefeitos seguem o sistema majoritário de votos: ganha o candidato que tiver a maior quantidade de votos [50]. Para deputados federais, deputados estaduais e vereadores, as eleições seguem o sistema proporcional: os votos não são computados para os candidatos, mas para seus partidos/coligações [49]. As eleições são divididas em fases [26, 39], sendo elas:

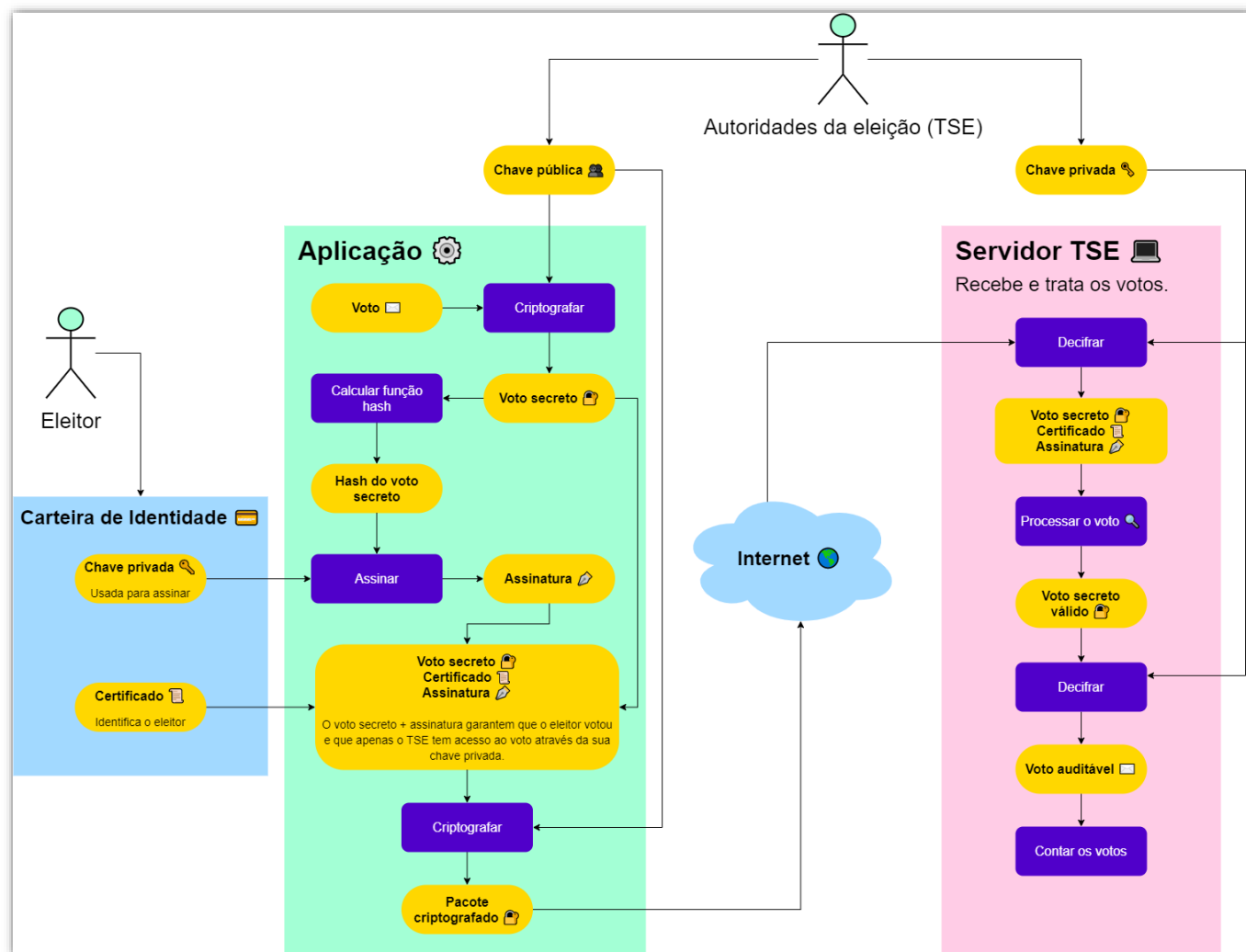
1. **Registro de candidatos:** cada partido ou coligação registra um candidato, seguindo alguns pré-requisitos.
2. **Cadastro de eleitores:** assim como os candidatos, os eleitores também precisam se registrar para poderem votar e receber o título de eleitor, um documento que prova a inscrição do eleitor.
3. **Logística eleitoral e preparação das eleições:** nesta fase ocorrem a manutenção e a verificação das peças essenciais da eleição, como transporte e distribuição das urnas, guarda e montagem das seções eleitorais, manutenção e testes das urnas etc.
4. **Votação:** é a etapa mais popular, que consiste na mobilização dos eleitores até as urnas para votarem em seus representantes.
5. **Prestação de contas:** os candidatos e os partidos prestam contas de acordo com a lei nº 9.504 de 1997 para garantir a transparência das eleições. Os candidatos eleitos cujas contas forem rejeitadas podem ser impedidos de tomarem posse de seus cargos.

6. **Totalização e divulgação dos resultados das eleições:** nesta parte, ocorre a contagem dos votos, descartando os votos nulos e brancos. Ela começa quando a votação é finalizada nas seções, então os boletins de urna são assinados e criptografados, para serem levados aos respectivos TREs (TSE no caso da eleição para Presidente da República).
7. **Diplomação dos candidatos eleitos:** por fim, depois dos resultados serem divulgados, a Justiça Eleitoral atesta a aptidão dos candidatos de assumirem seus cargos. O presidente do TSE assina e entrega o diploma, no caso do Presidente da República, e do TRE para os demais cargos.

Neste trabalho, a solução será demonstrada em um cenário de eleições digitais. Portanto, para uma demonstração simples, consideramos apenas a etapa de votação. As outras etapas estão fora do escopo deste trabalho.

Na eleição digital fictícia construída, o mecanismo de votação é composto de duas aplicações, uma para o eleitor e outra para o TSE (ou TRE). A aplicação do eleitor recebe o certificado público, a assinatura digital e o voto do eleitor, criptografa e salva os dados para que a aplicação do TSE possa recebê-los. A aplicação do TSE coleta os dados, descriptografar os dados, valida a assinatura digital e retorna o voto. Este sistema é ilustrado pela Figura 24.

Figura 24 – Funcionamento das eleições digitais



Fonte: Kauan Manzato do Nascimento, 2022.

Além disso, a aplicação do TSE tem à disposição o voto do eleitor e seu certificado contendo as informações do eleitor para que estas informações sejam tratadas para as etapas futuras da eleição, como a contagem de votos e a garantia que a pessoa só vote uma vez. Entretanto, como as fases posteriores estão fora do escopo, estas informações não serão tratadas neste momento.

4.2 – APLICAÇÃO DO ELEITOR

A aplicação do eleitor recebe o certificado digital do eleitor, vindo da carteira de identidade proposta, e pede para o eleitor inserir o PIN que protege a chave privada, autenticando a identidade do eleitor no lado do cliente. Inserindo o PIN correto, o aplicativo

carrega a chave privada e recebe o voto (um número inteiro) do eleitor. Depois, o voto é criptografado com a chave RSA do TSE e, a partir da chave privada do eleitor, é criada uma assinatura digital. Então, uma chave compartilhada é derivada usando ECDH que permite criptografar e enviar os dados em um meio inseguro. Os dados necessários para contabilizar o voto pela aplicação do TSE são criptografados usando a cifra AES e a chave derivada. Por fim, os dados são salvos localmente, simulando o envio dos dados por um meio inseguro, como a Internet. Em outras palavras, a aplicação segue os seguintes passos:

1. Carregar o certificado digital do eleitor no formato PEM
2. Carregar a chave privada do eleitor
3. Carregar as chaves públicas do TSE
4. Escolher o voto
5. Criptografar o voto
6. Gerar assinatura digital
7. Derivar a chave compartilhada
8. Criptografar os dados
9. “Enviar” os dados para o TSE

Passo 1: carregar o certificado digital do eleitor no formato PEM

O certificado é carregado usando a função `read_certificate` (Fig. 25). A função recebe o caminho do arquivo, lê os bites salvos e carrega no formato adequado, que é o padrão X.509 para certificados digitais.

Figura 25 – Definição da função `read_certificate`

```
def read_certificate(path):  
    with open(path, 'rb') as file:  
        cert_pem = file.read()  
    cert = x509.load_pem_x509_certificate(cert_pem)  
    return cert
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 2: carregar a chave privada do eleitor

Depois, a chave privada do eleitor é carregada usando a função `load_private_key` (Fig. 26). Esta função recebe o caminho da chave privada e permite três tentativas para inserir o PIN que protege a chave. Ao inserir o PIN, é chamada a função `get_private_key` (Fig. 27) que, além do caminho da chave, recebe o PIN. Se o PIN estiver correto, ela retorna a chave privada. Caso contrário, ela retorna um erro. Caso o usuário erre o PIN três vezes, a aplicação é fechada.

Figura 26 – Definição da função `load_private_key`

```
def load_private_key(path):
    for i in range(1,4):
        try:
            passwd = input('Insira seu PIN de 4 dígitos: ').encode()
            private_key = get_private_key(path, passwd)
            print('Chave carregada com sucesso.\n')
            return private_key
        except:
            print('PIN incorreto. Você tem mais', 3 - i, 'tentativas.\n')
    print('Você errou o PIN três vezes. Saindo da aplicação...')
    exit()
```

Fonte: Kauan Manzato do Nascimento, 2022.

Figura 27 – Definição da função `get_private_key`

```
def get_private_key(path, passwd):
    with open(path, 'rb') as file:
        private_key = serialization.load_pem_private_key(
            file.read(),
            password = passwd
        )
    return private_key
```

Fonte: Kauan Manzato do Nascimento, 2022.

A partir da chave privada, extraímos a chave pública usando o método `public_key` da chave privada, conforme Figura 28.

Figura 28 – Carregamento da chave privada

```
ct_private_key = load_private_key('./files/carteira/chave.pem')
ct_public_key = ct_private_key.public_key()
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 3: carregar as chaves públicas do TSE

As chaves públicas do TSE são importantes para que possamos mandar os dados criptografados para o TSE, sem que seja possível ver o conteúdo (confidencialidade). Fazemos isso com a função `read_public_key`, conforme Fig. 29:

Figura 29 – Definição da função `read_public_key`

```
def read_public_key(path):  
    with open(path, 'rb') as file:  
        return serialization.load_pem_public_key(file.read())
```

Fonte: Kauan Manzato do Nascimento, 2022.

A função `read_public_key` recebe o caminho para o arquivo da chave pública em formato PEM e retorna a chave pública. Dessa forma, carregamos as chaves RSA e EC do TSE (Fig. 30).

Figura 30 – Carregamento das chaves públicas do TSE

```
tse_rsa_public_key = read_public_key('./files/tse/rsa_public_key.pem')  
tse_ec_public_key  = read_public_key('./files/tse/ec_public_key.pem')
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 4: escolher o voto

Este passo também é simples, sendo realizado pela função `set_vote`. Esta função não recebe argumentos, pede para o usuário inserir o voto, neste caso um número inteiro, e retorna o valor em bytes, para que possa ser criptografado mais tarde (Fig. 31).

Figura 31 – Definição da função `set_vote`

```
def set_vote():  
    vote = int(input('Escolha o seu voto: '))  
    print('Seu voto é {}'.format(vote))  
    return vote.to_bytes(length = 3, byteorder = 'big')
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 5: criptografar o voto

O voto, em bytes, é criptografado com a chave pública RSA de 4096 bits do TSE por meio da função `encrypt` da chave pública, disponibilizada pela biblioteca `cryptography`, assim os dados só serão acessíveis com a chave privada do TSE e ilegíveis para possíveis agentes maliciosos (Fig. 32).

Figura 32 – Criptografia do voto

```
voto_secreto = tse_rsa_public_key.encrypt(
    voto,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA384()),
        algorithm=hashes.SHA384(),
        label=None
    )
)
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 6: gerar assinatura digital

A assinatura digital é feita com a função `sign` das chaves privadas da biblioteca `cryptography`. Esta função recebe os dados que serão assinados em bytes e recebe o método de assinatura, neste caso o ECDSA com o hash SHA-384 (Fig. 33).

Figura 33 – Criação da assinatura digital

```
assinatura = ct_private_key.sign(
    voto_secreto,
    ec.ECDSA(hashes.SHA384())
)
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 7: derivar a chave compartilhada

Esse passo é mais complexo porque trabalha com várias informações diferentes. Primeiro, é preciso saber a importância desse passo: ao derivarmos uma chave compartilhada, podemos estabelecer uma conexão segura entre eleitor e TSE, porque criptografaremos com uma chave que só essas duas partes conhecem. O segundo ponto importante é que derivamos a

chave usando o algoritmo ECDH, combinando a chave privada de quem está derivando a chave e a chave pública da outra parte envolvida comunicação e o vetor inicial. Por fim, a chave derivada e o vetor inicial são usados na cifra simétrica AES para cifrar e decifrar as informações.

O ECDH foi implementado na função `derive_key`, que recebe a chave privada de quem está derivando e a chave pública da outra parte. A função retorna, em bytes, a chave compartilhada que foi derivada (Fig. 34).

Figura 34 – Definição da função `derive_key`

```
# ECDH (troca de chaves)
def derive_key(your_private_key, their_public_key):
    shared_key = your_private_key.exchange(ec.ECDH(), their_public_key)
    return HKDF(
        algorithm=hashes.SHA384(),
        length=32,
        salt=None,
        info=None
    ).derive(shared_key)
```

Fonte: Kauan Manzato do Nascimento, 2022.

Na aplicação do eleitor, a função `derive_key` recebe a chave privada do eleitor e a chave pública do TSE (Fig. 35). Essa mesma função será usada na aplicação do TSE para derivar a mesma chave e recuperar as informações cifradas.

Figura 35 – Derivação da chave compartilhada usando ECDH

```
derived_key = derive_key(ct_private_key, tse_ec_public_key)
```

Fonte: Kauan Manzato do Nascimento, 2022.

Um ponto a ressaltar é que a comunicação segura podia ser feita usando a chave RSA do TSE, a mesma usada por tornar o voto secreto. Entretanto, é mais seguro usar chaves separadas para funções diferentes, o ECDH é muito usado atualmente e, portanto, importante demonstrarmos aqui também, e é interessante mostrar que a criptografia de chave pública permite uma mesma operação de várias formas diferentes, neste caso a comunicação segura entre duas entidades.

Passo 8: criptografar os dados

A chave derivada e compartilhada vai ser usada pela cifra AES para criptografar os dados e enviá-los para o TSE de forma segura. Primeiro, geramos o vetor inicial (IV), que são 16 bytes aleatórios usados pela cifra AES no modo GCM. Esta informação é o ponto de partida para a cifra e não precisa ser secreto, por isso não criptografaremos o IV, simulando o fato que as duas entidades concordaram com o IV. O IV é gerado conforme Fig. 36.

Figura 36 – Criação do vetor inicial IV

```
from secrets import token_bytes  
  
iv = token_bytes(16)
```

Fonte: Kauan Manzato do Nascimento, 2022.

Depois, passamos os dados, a chave compartilhada e o IV para a função `aes_encrypt_data`, responsável por cifrar os dados com a chave, e cuja definição é dada conforme Figura 37.

Figura 37 – Definição da função `aes_encrypt_data`

```
def aes_encrypt_data(data, key, iv):  
    aes = Cipher(algorithms.AES(key), modes.CBC(iv))  
    encryptor = aes.encryptor()  
  
    padder = sympadding.PKCS7(128).padder()  
    padded_data = padder.update(data) + padder.finalize()  
    return encryptor.update(padded_data) + encryptor.finalize()
```

Fonte: Kauan Manzato do Nascimento, 2022.

A função `aes_encrypt_data` recebe os dados, a chave e o vetor inicial. A função inicia a cifra, realiza o padding dos dados, já que a cifra funciona com blocos, e retorna os dados criptografados em bytes. Essa função, no código do eleitor, é usada para criptografar os dados a serem enviados para o TSE, conforme mostrado na figura 38.

Figura 38 – Criptografia dos dados a serem enviados para o TSE

```
enc_voto_secreto = aes_encrypt_data(voto_secreto, derived_key, iv)  
enc_assinatura = aes_encrypt_data(assinatura, derived_key, iv)  
enc_cert = aes_encrypt_data(cert_pem, derived_key, iv)
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 9: “enviar” os dados para o TSE

Por fim, salvamos os arquivos localmente como forma de simular o envio deles para o TSE. Para o propósito deste trabalho, o meio por onde esses dados passam é irrelevante, desde que possa ser considerado inseguro. A função responsável por salvar os dados é chamada de `save_file` (Fig. 39). A função `save_file` recebe os dados, em bytes, a serem salvos e o nome do arquivo, e salva os arquivos na pasta de envio para a aplicação do TSE utilizar depois.

Figura 39 – Definição da função `save_file`

```
def save_file(data, filename):  
    with open('./files/envio/'+filename, 'wb') as file:  
        file.write(data)
```

Fonte: Kauan Manzato do Nascimento, 2022.

E aqui terminamos a análise do código da aplicação do eleitor. Conseguimos mostrar que, com algumas linhas de código, é possível autenticar sua identidade com a assinatura e ainda proteger seus dados com a criptografia de chave pública. O voto, por exemplo, foi criptografado duas vezes, uma com a chave RSA e outra com a chave derivada do ECDH.

Na próxima seção, mostraremos como o TSE pode tratar as informações que chegam e como pode garantir a identidade do proprietário do certificado digital, dada a assinatura, idêntico ao que foi exposto na revisão bibliográfica.

4.3 – APLICAÇÃO DO TSE

A aplicação do TSE recebe as informações enviadas pela aplicação do cliente (eleitor), que inclui a chave pública, o certificado, a assinatura digital e o voto secreto do eleitor, todos criptografados e carrega as chaves privadas que serão usadas nas operações de criptografia posteriormente. Com a chave privada EC do TSE carregada é possível derivar a mesma chave usada pelo eleitor com o ECDH. Então, esta chave derivada e compartilhada é usada para decifrar os dados recebidos pelo eleitor. Por fim, a assinatura é verificada. Caso a assinatura seja válida, o voto é decifrado e processado. Caso contrário, o voto pode ser descartado. Em outras palavras, a aplicação segue os seguintes passos:

1. Carregar os arquivos que foram recebidos pelo eleitor

2. Carregar a chave pública do eleitor
3. Carregar as chaves privadas do TSE (EC e RSA)
4. Derivar a chave compartilhada entre TSE e eleitor com ECDH
5. Decifrar os dados recebidos pelo eleitor e carregar o certificado digital
6. Validação da assinatura e processamento do voto secreto

Passo 1: carregar os arquivos que foram recebidos pelo eleitor

Os arquivos que foram recebidos pelo cliente são carregados através da função `load_file`, que recebe o caminho do arquivo, lê o arquivo e retorna o seu conteúdo em bytes (Fig. 40). Os arquivos carregados incluem o voto secreto, a assinatura e o certificado digital criptografados, e o vetor inicial `iv`, usado na cifra AES (Fig. 41).

Figura 40 – Definição da função `load_file`

```
def load_file(path):  
    with open(path, 'rb') as file:  
        return file.read()
```

Fonte: Kauan Manzato do Nascimento, 2022.

Note que, na verdade, os arquivos não foram enviados através da Internet, mas apenas salvos localmente para fins de demonstração. Esse ponto não muda a conclusão do trabalho porque o meio em que os arquivos são passados não muda o fato que estão criptografados com a chave pública do TSE e só é possível de visualizá-los com a chave privada do TSE, que é secreta.

Figura 41 – Carregamento dos dados enviados pelo eleitor

```
# caminho das informações enviadas  
path_envio = './files/envio/'  
path_tse   = './files/tse/'  
  
# carregar os arquivos  
enc_voto_secreto = load_file(path_envio + 'enc_voto_secreto')  
enc_assinatura   = load_file(path_envio + 'enc_assinatura')  
cert_pem         = load_file(path_envio + 'certificado.pem')  
iv               = load_file(path_envio + 'iv')
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 2: carregar a chave pública do eleitor

A chave pública do eleitor é carregada com a função `read_public_key`, semelhante à função `load_file`. A diferença é que a função `read_public_key` recebe o caminho da chave pública, lê o conteúdo e retorna a chave no formato apropriado para ser usada (Fig. 42).

Figura 42 – Definição da função `read_public_key`

```
def read_public_key(path):  
    with open(path, 'rb') as file:  
        return serialization.load_pem_public_key(file.read())
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 3: carregar as chaves privadas do TSE (EC e RSA)

Esse passo é feito com a função `read_private_key`. Essa função recebe dois parâmetros: o caminho do arquivo onde a chave privada está armazenada e a senha da chave privada, ambos em bytes. A função `read_private_key` se baseia na função `load_pem_private_key` da biblioteca `cryptography`, que carrega a chave privada no formato PEM (Fig. 43).

Figura 43 – Definição da função `read_private_key`

```
# dado o arquivo e a senha, a função retorna a chave privada  
def read_private_key(path, passwd):  
    with open(path, 'rb') as file:  
        private_key = serialization.load_pem_private_key(  
            file.read(),  
            password = passwd  
        )  
    return private_key
```

Fonte: Kauan Manzato do Nascimento, 2022.

Aqui é importante ressaltar que essa função não é comprovadamente segura. Apesar da biblioteca `cryptography` ter aplicações reais, o mal uso das funções disponibilizadas por ela pode comprometer sua segurança. O objetivo deste cenário é apenas demonstrar como um certificado digital pode ser usado para autenticar a identidade dos eleitores em um cenário fictício de eleições digitais.

Voltando ao código, a função `read_private_key` é usada para carregar a chave privada EC e RSA do TSE. A chave privada EC é usada pelo ECDH para decifrar os dados recebidos e a chave privada RSA é usada para decifrar o voto secreto (Fig. 44).

Figura 44 – Carregamento das chaves privadas do TSE

```
tse_ec_private_key = read_private_key(path_tse + 'ec_private_key.pem', b'9999')
tse_rsa_private_key = read_private_key(path_tse + 'rsa_private_key.pem', b'9999')
```

Fonte: Kauan Manzato do Nascimento, 2022.

Perceba que as senhas estão embutidas no código, o que é, obviamente, uma má prática de programação, além de ser uma vulnerabilidade. Entretanto, como o objetivo desta aplicação não é ser a solução mais segura, mas a demonstração da carteira digital, podemos ignorar esta vulnerabilidade para mantermos a simplicidade da aplicação.

Passo 4: derivar a chave compartilhada entre TSE e eleitor com ECDH

Este passo é realizado através da mesma função `derive_key` que foi definida na aplicação do cliente. Na aplicação do TSE, esta função recebe a chave privada do TSE (quem está derivando) e a chave pública do eleitor (Fig. 45). A chave derivada é usada para decifrar os dados recebidos.

Figura 45 – Derivação da chave compartilhada usando ECDH

```
derived_key = derive_key(tse_ec_private_key, ct_public_key)
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 5: decifrar os dados recebidos pelo eleitor e carregar o certificado digital

Os dados do eleitor carregados são decifrados usando a cifra AES no modo GCM, implementada pela função `aes_decrypt_data`, semelhante à função `aes_encrypt_data` usada pelo eleitor para encriptar os dados. A função para decifrar recebe os dados em bytes, a chave compartilhada e o vetor inicial (Fig. 46).

Figura 46 – Definição da função `aes_decrypt_data`

```
def aes_decrypt_data(data, derived_key, iv):  
    aes = Cipher(algorithms.AES(derived_key), modes.CBC(iv))  
    decryptor = aes.decryptor()  
  
    decrypted_data = decryptor.update(data) + decryptor.finalize()  
    unpadder = sympadding.PKCS7(128).unpadder()  
    return unpadder.update(decrypted_data) + unpadder.finalize()
```

Fonte: Kauan Manzato do Nascimento, 2022.

Com a função `aes_decrypt_data`, deciframos o voto secreto, a assinatura e o certificado criptografados. Adicionalmente, o certificado digital é convertido para o formato apropriado usando a função `load_pem_x509_certificate` (Fig. 47).

Figura 47 – Deciframento dos dados enviados pelo eleitor

```
voto_secreto = aes_decrypt_data(enc_voto_secreto, derived_key, iv)  
assinatura = aes_decrypt_data(encassinatura, derived_key, iv)  
cert_pem = aes_decrypt_data(enc_cert_pem, derived_key, iv)  
  
cert = x509.load_pem_x509_certificate(cert_pem)
```

Fonte: Kauan Manzato do Nascimento, 2022.

Passo 6: validação da assinatura e processamento do voto secreto

Esse último passo é importante porque é o que queremos mostrar neste trabalho, que é a autenticação da identidade digitalmente. Conforme dito na revisão da bibliografia, uma assinatura digital válida significa a autenticidade da identidade de uma das partes da comunicação. Em outras palavras, se a assinatura digital for válida para um dado certificado digital, então aquela pessoa é quem realmente diz ser. Assim, é garantido que a mensagem não foi violada e só pode ter sido criada pelo proprietário do certificado, o eleitor.

Para verificar a assinatura, foi implementada a função `verificar_assinatura`, que faz uso da função `verify` oferecida pela biblioteca `cryptography`. A função `verificar_assinatura` recebe a chave pública do eleitor, a assinatura digital e os dados e tenta validar a assinatura com a função `verify`, que recebe a assinatura, os dados e usa o ECDSA para fazer a verificação. Se a assinatura for válida, a função `verificar_assinatura` retorna `False`. Caso contrário, retorna `True` (Fig. 48).

Figura 48 – Definição da função `verificar_assinatura`

```
def verificar_assinatura(public_key, signature, data):  
    is_valid = False  
    try:  
        public_key.verify(signature, data, ec.ECDSA(hashes.SHA384()))  
        is_valid = True  
    except:  
        pass  
    return is_valid
```

Fonte: Kauan Manzato do Nascimento, 2022.

No código, a função `verificar_assinatura` recebe a chave pública do eleitor, a assinatura digital e os dados que foram assinados, que no caso é o voto secreto, e retorna um valor booleano usado em uma condição. Se a assinatura digital for válida, o voto pode ser considerado válido e contabilizado. Caso contrário, o voto pode ser descartado (Fig. 49).

Figura 49 – Validação da assinatura digital e processamento básico do voto

```
validade_assinatura = verificar_assinatura(  
    public_key = ct_public_key,  
    signature = assinatura,  
    data = voto_secreto  
)  
  
if(validade_assinatura):  
    voto = decrypt_vote(voto_secreto, tse_rsa_private_key)  
    print('Bem-vindo ' + get_cert_subject(cert))  
    print('Sua carteira de identidade foi emitida por {}'.format(get_cert_issuer(cert)))  
    print('Seu voto é válido e foi contabilizado!')  
    print('Você votou em {}'.format(voto))  
else:  
    print('Certificado inválido. Voto descartado.')
```

Fonte: Kauan Manzato do Nascimento, 2022.




Com a assinatura válida, o voto secreto é decifrado com a função `decrypt_vote`, que recebe o voto secreto (criptografado com RSA) e a chave privada RSA do TSE. A função `decrypt_vote` usa a função `decrypt` da chave privada RSA e retorna o voto como um inteiro.

Como o certificado do eleitor está disponível, é possível saber em quem o eleitor votou, o que não é o ideal. Estes dois dados podem ser tratados separadamente, mas, assim como a leitura das chaves privadas do TSE, este não é o objetivo do trabalho e, portanto, optamos pela simplicidade, tratando-os em conjunto. Na próxima seção, o código será executado para demonstrar o resultado.

4.4 DEMONSTRAÇÃO DAS ELEIÇÕES DIGITAIS





Antes de executar as aplicações, algumas informações precisaram ser criadas previamente para poderem ser usadas nas eleições. Dentre as informações criadas antes da execução, tem a carteira de identidade digital (Fig. 50), que foi criada usando o código de emissão de carteira digital e que estão dentro da pasta `./files/carteira`, e tem as chaves do TSE usadas (EC e RSA), que estão na pasta `./files/tse` (Fig. 51).

Figura 50 – Arquivos da carteira de identidade digital

Nome	Tipo	Tamanho
 certificado.pem	CMS (S/MIME) File	1 KB
 chave.pem	CMS (S/MIME) File	1 KB
 nome	Arquivo	1 KB

Fonte: Kauan Manzato do Nascimento, 2022.

Figura 51 – Chaves usadas pelo TSE

Nome	Tipo	Tamanho
 ec_private_key.pem	CMS (S/MIME) File	1 KB
 ec_public_key.pem	CMS (S/MIME) File	1 KB
 rsa_private_key.pem	CMS (S/MIME) File	4 KB
 rsa_public_key.pem	CMS (S/MIME) File	1 KB

Fonte: Kauan Manzato do Nascimento, 2022.

Com tudo configurado, executamos a aplicação do eleitor com algum interpretador de Python. Neste caso, o código foi executado usando o terminal do Windows, como mostrado na Figura 52.

Assim como o código de emissão da carteira de identidade digital, o código das eleições digitais pode ser executado com qualquer interpretador de Python. Neste caso, vamos executar o código pelo terminal do Windows.

Figura 52 – Execução da aplicação do eleitor (parte 1)

```
Carregando seu certificado digital...
Certificado carregado.

*****
*                               *
*  ELEIÇÕES 2042               *
*                               *
*****

Seja bem-vindo(a) às eleições digitais de 2042, Sr(a). Kauan Manzato do Nascimento

Para prosseguir com a votação, precisamos validar a sua identidade.
Insira seu PIN de 4 dígitos: 3011
```

Fonte: Kauan Manzato do Nascimento, 2022.

A aplicação lê o certificado digital do eleitor, dá as boas-vindas ao eleitor e pede o PIN para confirmar a identidade do indivíduo (Fig. 52). Com o PIN correto, a chave privada é carregada, junto com as chaves públicas do TSE, e então a aplicação pede para o usuário inserir o voto (Fig. 53).

Figura 53 – Execução da aplicação do eleitor (parte 2)

```
Escolha o seu voto: 123
Seu voto é 123

Criptografando seu voto...
O voto foi criptografado com sucesso.

Assinando seu voto...
Assinatura digital feita com sucesso.






Criando conexão segura com o TSE...
Criptografando os dados...
Enviando os dados...

Pronto. Seu voto foi enviado ao TSE. Agora é só esperar os resultados.
PS D:\UFABC\PGC FINAL\Código\eleições> |
```

Fonte: Kauan Manzato do Nascimento, 2022.

Com o voto escolhido, a aplicação criptografa o voto com RSA, cria a assinatura digital, criptografa tudo com auxílio do ECDH e salva os arquivos na pasta `./files/envio`, simulando o envio dos dados em um meio inseguro de comunicação (Fig. 54). Os dados com o prefixo `enc` são os dados criptografados, composto de bytes ilegíveis, já a `chave_publica_eleitor.pem` e o `iv` estão em texto plano.

Figura 54 – Arquivos criados pela aplicação do eleitor

Nome	Tipo	Tamanho
 chave_publica_eleitor.pem	CMS (S/MIME) File	1 KB
 encassinatura	Arquivo	1 KB
 enc_cert	Arquivo	1 KB
 enc_voto_secreto	Arquivo	1 KB
 iv	Arquivo	1 KB

Fonte: Kauan Manzato do Nascimento, 2022.

Agora, com os dados “enviados” ao TSE, executamos a aplicação do TSE para tratar o voto, também usando o terminal do Windows (Fig. 55). Como não é necessário qualquer entrada de informações por parte do usuário, a aplicação executa sozinha até o final, conforme descrita anteriormente.

Figura 55 – Execução da aplicação do TSE

```
PS D:\UFABC\PGC FINAL\Código\eleições> python.exe .\tse.py
Bem-vindo Kauan Manzato do Nascimento
Sua carteira de identidade foi emitida por Cartorio de Santo Andre
Seu voto é válido e foi contabilizado!
Você votou em 123
PS D:\UFABC\PGC FINAL\Código\eleições> |
```

Fonte: Kauan Manzato do Nascimento, 2022.

Caso algum byte dos dados enviados seja modificado indevidamente, as operações de decifrar os dados e validação da assinatura podem falhar. Por exemplo, ao modificarmos um único byte do voto secreto criptografado (**enc_voto_secreto**) (Fig. 56), a verificação da assinatura falha porque os dados assinados estão modificados (Fig. 57). Portanto, para que o voto seja contabilizado, é preciso uma combinação de autenticação, confidencialidade e integridade.

Figura 56 – Falha na verificação do certificado

```
PS D:\UFABC\PGC FINAL\Código\eleições> python.exe .\tse.py
Certificado inválido. Voto descartado.
```

Fonte: Kauan Manzato do Nascimento, 2022.

Figura 57 – Modificação do primeiro byte do voto secreto, usando o programa HxD

```

HxD - [D:\UFABC\PGC FINAL\Código\eleições\files\envio\enc_voto_secreto]
Arquivo Editar Localizar Exibir Análise Extras Janelas Ajuda
16 Windows (ANSI) hex
enc_voto_secreto
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F Texto decodificado.
00000000 00 30 2B 14 2B E0 C7 F4 4B 67 74 9E 57 7B E1 49 .[+.+âÇOKgtzW(âI
00000010 6F 4A 56 C8 C1 1A C4 E7 46 FB 6E 0B C1 21 51 1A oJVEÄ.ÄçFûn.Ä!Q.
00000020 99 55 92 8A 11 4F 99 F0 0C ED 73 15 58 36 30 40 "U'Š.O"š.is.X60@
00000030 EC DB 56 71 C5 14 EC 70 90 1A DE 98 22 8A 31 A7 iÜVqÄ.ip..B"Šis
00000040 24 67 FD C3 2E 06 2F DC 2F 64 52 83 E5 F8 7B 78 $gyÄ../Ü/dRfâø{x
00000050 45 1A F6 C6 18 52 1C 48 49 3E 5D 14 79 34 73 CC E.öÄ.R.HI>].y4sI
00000060 37 90 BB F6 25 43 4A D1 04 96 90 5C 73 D5 24 5F 7.»ö%ÇJŇ.-.\sÖš
00000070 47 1D 50 FA EA 40 50 E8 9D 17 DE 79 98 D4 D6 F2 G.Püê@Pè..By"ÖÖö
00000080 C0 11 74 1D D3 BE 6C 2B F6 16 D6 DE 1D 8E C8 88 Ä.t.Ö*1+ö.ÖP.ZÊ^
00000090 5D 17 9D 40 00 22 A2 EC 17 52 32 AD 79 B0 B0 F6 ].@."ci.R2.y°ö
000000A0 29 AF 7E 57 C8 48 64 99 9E 29 7F EA 5B A6 7D 8E )~WEHd™ž).ê[;]ž
000000B0 FD DC 9B 30 3A 0D 26 7E 7B 87 54 22 3F 8D B1 08 ýÜ>0:..&~{+T"?..±.
000000C0 5A BA F7 55 51 53 B7 0A C8 E0 D2 BC AD 2D B8 8C Z°>UQS..ÈäÖ4.-.œ
000000D0 45 5F D2 1C 80 57 10 AE AD 40 2B 4E B2 33 61 ED E.Ö.€W.ö.ö+N°3ai
000000E0 D5 81 EF C7 A4 90 5C 49 0D F7 F7 77 AE A6 82 5E Ö.iÇ#.\\I.÷÷w@!,^
000000F0 52 29 A8 D1 AB 95 93 6E 0B 92 84 BA D9 64 12 AD R)"Ň*."n.'."°Üd..
00000100 48 C8 7A 17 DA 20 87 D1 52 9C B3 9A 00 C6 B7 04 HÊz.Ü #ŇRœ°š.Æ..
00000110 E3 5F 6F 87 31 2E E1 A8 AF 6F BA 5E 58 C7 8C 81 ä_o+1.á""o°^XÇE.
00000120 A4 A5 5B C7 DF 9B C5 E8 AC AA 82 73 B9 2A 97 EC HŸ[ÇB>Äê-+.s°*-i
00000130 DC F4 45 D6 34 D1 3D 55 36 3C 36 C1 D3 C5 F8 26 ÜöEÖ4Ň=U6<6ÄÖÄö&
00000140 61 7C 0F 20 FD 13 3E B1 B6 BA 32 09 57 1A EA 86 a|.ý.>±Ÿ°2.W.êt
00000150 7A C8 43 1B CD 69 BE 74 92 D1 D9 20 17 5B 2E 8D zÊC.íi%t'ŇÜ.[..
00000160 61 27 54 35 76 38 FB D5 3E DD C9 E0 EB A0 FE 3D a'T5v8üÖ>ŸÊäè p=
00000170 A3 BC CF 5E BD 1B 25 3E 0F 64 C3 81 69 67 DB C9 £4I^s.š>.dÄ.igÜÊ
00000180 C4 5D 33 B4 E3 F9 7D 06 2F 12 E2 77 E9 38 4F 2E Ä]3'äù)./.âwé80.
00000190 99 EA B7 79 48 15 14 EC 1B 51 39 4F 8A A9 90 39 "è.yH..i.Q90Šö.9
000001A0 0B DD B9 3C 8B DE 15 7F EF DC 84 B2 3F 10 B7 FE .Ÿ°<<P..iÜ..°?.p
000001B0 BB 89 75 0E 20 EF 7E 38 57 10 E9 27 6F 26 E7 BD »ku. i~8W.é'o&çs
000001C0 2C 95 3B 78 79 A0 06 8A 32 C9 CB 3D 9E B3 D2 EA .,;xy .Š2ÊÊ=ž°Öè
000001D0 D8 75 02 9F 62 E9 3A 69 77 41 55 B9 D8 32 20 75 ōu.Ÿbè:iwAU°02 u
000001E0 39 A9 8E 32 89 F4 2D F7 51 5A 23 55 31 26 FD CC 9öŽ2%ö-÷QZ#U1&yİ
000001F0 5A 5C E8 37 3B 83 EF 24 D5 63 3A ED BA 3D E0 EE Z\è7;fišÖc:i°=âi
00000200 0E DE BA D0 69 32 FC BB 11 A0 5A B8 BB AA 5D 78 .P°Đi2ü». Z.»°]x
  
```

Fonte: Kauan Manzato do Nascimento, 2022.

Com isso, terminamos a análise e a demonstração do código das eleições digitais. Conseguimos demonstrar que, mesmo com poucas linhas de código, a aplicação foi capaz de gerar dados confidenciais e autenticar a identidade do proprietário do certificado, evitando fraudes. Esse cenário é um bom demonstrativo de como a criptografia de chave pública é poderosa, simples de ser implementada e transparente, no sentido de que é possível tornar o código aberto para auditorias de qualquer cidadão e ainda manter a segurança.

Uma última observação, mas não menos importante, é o fato de que essa aplicação é apenas uma demonstração do conteúdo explicado neste trabalho, que é uma alternativa à autenticação de identidade que temos atualmente no Brasil, e não uma aplicação real. O código está propenso a vulnerabilidades, como, por exemplo, o fato de trabalharmos com uma chave para autenticação e assinatura, e não uma chave para cada função, e, por isso, não é recomendada seu uso em cenários reais.

5 – CONCLUSÃO

Este trabalho revisou a bibliografia a fim de concentrar os conceitos essenciais para o desenvolvimento da solução proposta, foram dadas as causas do problema de autenticação da identidade no Brasil e as consequências, os quais justificam a criação da solução proposta, e foi criado um cenário fictício de eleições digitais para implementar a solução proposta e demonstrar como ela pode ser aplicada em cenários reais e suas vantagens. Além disso, o trabalho mostrou outros países que já implementam essa solução, como a Estônia, e como a solução impacta positivamente a população.

Com este trabalho, foi demonstrado que a criptografia de chave pública tem um grande potencial para mudar a forma como nos identificamos, como nos comunicamos e até como votamos. De fato, as aplicações da criptografia são muitas, mas aqui focamos em apenas uma: autenticação da identidade no contexto brasileiro. Também foi demonstrado que a solução é fácil de ser implementada e segura, além do fato de ser transparente: qualquer cidadão pode verificar os mecanismos de segurança, porque são todos baseados na matemática, e não em segurança através da obscuridade.

Os próximos passos para desenvolver a solução ainda mais consiste em estudar os pontos negativos da solução, aprofundar o conhecimento sobre os custos de implementação de carteiras físicas de identidade e comparar com a solução atual, demonstrar outras formas de autenticação, como a autenticação TLS, estudar os impactos sociais da solução e emitir um cartão inteligente real para demonstrar a solução em outros cenários práticos.

6 – REFERÊNCIAS BIBLIOGRÁFICAS

- [1] ABRAMS, M. et al. **Guide to Industrial Control Systems (ICS) Security**. National Institute of Standards and Technology, 2015. Disponível em: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>. Acesso em: 28 abr. 2022.

- [2] ADLEMAN, L.; RIVEST, R.; SHAMIR, A. **A Method for Obtaining Digital Signatures and Public-Key Cryptosystems**. Communications of the ACM, n. 21, p. 120-126, 1978. Disponível em: <https://people.csail.mit.edu/rivest/Rsapaper.pdf>. Acesso em: 10 fev. 2022.

- [3] ALEMANHA. **Erste, Zweite und Dritte Bekanntmachung über den Kennkartenzwang**. Deutsches Reichsgesetzblatt, 23 jul. 1938, vol. 1938, n.115, p. 921-922. Disponível em: https://de.wikisource.org/wiki/Bekanntmachungen_%C3%BCber_den_Kennkartenzwang. Acesso em: 09 mar. 2022.

- [4] ANDRETTA, F.; ARAÚJO, C. **Mentir para receber os R\$ 600 é fraude e pode dar mais de 6 anos de prisão**. UOL Economia, [S. l.], 4 jun. 2020. Disponível em: <https://economia.uol.com.br/noticias/redacao/2020/06/04/auxilio-emergencial-crime-fraude-estelionato-r-600.htm>. Acesso em: 15 jul. 2021.

- [5] ÁVILA, C. S.; MARTÍNEZ, V. G. **Analysis of ECIES and other Cryptosystems based on Elliptic Curves**. International Journal of Information Assurance and Security. n. 6. p. 1-9, fev. 2011.

- [6] BÉLGICA. **What is the eID?** eID software. Disponível em: <https://eid.belgium.be/en/what-eid>. Acesso em: 15 abr. 2022.

- [7] BRASIL. **Decreto-lei nº 2.848, de 7 de dezembro de 1940**. Rio de Janeiro. Disponível em: http://www.planalto.gov.br/ccivil_03/decreto-lei/del2848.htm. Acesso em: 14 jul. 2021.

- [8] BRASIL. **Lei nº 4.862, de 29 de novembro de 1965**. Brasília. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/L4862.htm. Acesso em: 15 jul. 2021.
- [9] BRASIL. **Lei nº 7.116, de 29 de agosto de 1983**. Assegura validade nacional às Carteiras de Identidade, regula sua expedição e dá outras providências. Brasília. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/1980-1988/17116.htm. Acesso em: 29 jul. 2021.
- [10] BRASIL. Constituição (1988). **Constituição da República Federativa do Brasil de 1988**. Brasília, DF. Senado Federal, 1988. Disponível em: http://www.planalto.gov.br/ccivil_03/constituicao/constituicaocompilado.htm. Acesso em: 21 abr. 2022.
- [11] BRASIL. Câmara dos Deputados. **Projeto de Lei nº 496, de 1995**. Dispõe sobre o registro civil e o documento único de identificação da pessoa natural em todo o território nacional e dá outras providências. Disponível em: https://www.camara.leg.br/proposicoesWeb/prop_mostrarintegra?codteor=1134522&filename=Dossie+-PL+496/1995. Acesso em 25 out. 2021.
- [12] BRASIL. **Lei nº 9.454, de 7 de abril de 1997**. Institui o número único de Registro de Identidade Civil e dá outras providências. Brasília. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/19454.htm. Acesso em: 14 jul. 2021.
- [13] BRASIL. **Medida provisória nº 2.200-2, de 24 de agosto de 2001**. Institui a Infraestrutura de Chaves Públicas Brasileira - ICP-Brasil, transforma o Instituto Nacional de Tecnologia da Informação em autarquia, e dá outras providências. Disponível em: http://www.planalto.gov.br/ccivil_03/MPV/Antigas_2001/2200-2.htm. Acesso em: 15 abr. 2022.
- [14] BRASIL. **Decreto nº 7.166, de 5 de maio de 2010**. Cria o Sistema Nacional de Registro de Identificação Civil, institui seu Comitê Gestor, regulamenta disposições da Lei no 9.454, de 7 de abril de 1997, e dá outras providências. Brasília. Disponível em: http://planalto.gov.br/ccivil_03/_ato2007-2010/2010/Decreto/D7166.htm. Acesso em: 15 jul. 2021.

- [15] BRASIL. Congresso Nacional. **Projeto de lei 1775, de 2015**. Dispõe sobre o Registro Civil Nacional - RCN e dá outras providências. Disponível em: https://www.camara.leg.br/proposicoesWeb/prop_mostrarintegra?codteor=1342951. Acesso em: 25 out. 2021.
- [16] BRASIL. Congresso Nacional. **Lei nº 13.444, de 11 de maio de 2017**. Dispõe sobre a Identificação Civil Nacional (ICN). Brasília. Disponível em: http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2017/lei/113444.htm. Acesso em: 25 out. 2021.
- [17] BRASIL. **Decreto nº 9.278, de 5 de fevereiro de 2018**. Regulamenta a Lei nº 7.116, de 29 de agosto de 1983, que assegura validade nacional às Carteiras de Identidade e regula sua expedição. Brasília. Disponível em: http://www.planalto.gov.br/ccivil_03/_Ato2015-2018/2018/Decreto/D9278.htm. Acesso em: 25 out. 2021.
- [18] BRASIL. **Decreto nº 10.046, de 9 de outubro de 2019**. Dispõe sobre a governança no compartilhamento de dados no âmbito da administração pública federal e institui o Cadastro Base do Cidadão e o Comitê Central de Governança de Dados. Brasília. Disponível em: http://www.planalto.gov.br/ccivil_03/_ato2019-2022/2019/decreto/D10046.htm. Acesso em: 25 out. 2021.
- [19] BRASIL. Congresso Nacional. **Projeto de lei 1.422, de 2019**. Institui o Cadastro de Pessoas Físicas (CPF) como número suficiente para identificação do cidadão nos bancos de dados de serviços públicos, altera dispositivos da Lei nº 13.460, de 26 de junho de 2017, e dá outras providências. Brasília. Disponível em: https://www.camara.leg.br/proposicoesWeb/prop_mostrarintegra;jsessionid=node0hc634df4zwgpb2a5uedtixl4675198.node0?codteor=1718365&filename=PL+1422/2019. Acesso em: 15 jul. 2021.
- [20] BRASIL. **Lei nº 13.982, de 2 de abril de 2020**. Brasília. Disponível em: http://www.planalto.gov.br/ccivil_03/_ato2019-2022/2020/lei/113982.htm. Acesso em: 14 jul. 2021.

- [21] BYBEE, H. C.; HOUZE, A. **Nineteenth-Century French Passport Laws and Documents**. The BYU Family Historian, 01 set. 2007, vol. 6. Disponível em: <https://scholarsarchive.byu.edu/cgi/viewcontent.cgi?article=1030&context=byufamilyhistorian>. Acesso em: 09 mar. 2022.
- [22] CAIXA ECONÔMICA FEDERAL. **Auxílio Emergencial 2021**. CAIXA. Disponível em: <https://www.caixa.gov.br/auxilio/auxilio2021/Paginas/default.aspx>. Acesso em: 14 jul. 2021.
- [23] CAZAQUISTÃO. **Passport and Identity Card of the RK**. eGov. Disponível em: <https://web.archive.org/web/20220122201157/http://egov.kz/cms/en/categories/passport>. Acesso em: 15 abr. 2022.
- [24] COOK, J. D. **Elliptic curve P-384**. 11 mai. 2019. Disponível em: <https://www.johndcook.com/blog/2019/05/11/elliptic-curve-p-384/>. Acesso em: 17 fev. 2022.
- [25] DIFFIE, W.; HELLMAN, M. **New directions in cryptography**. IEEE Transactions on Information Theory, vol. IT-22, n. 6, nov. 1976. Disponível em: <https://ieeexplore.ieee.org/document/1055638/>. Acesso em: 05 abr. 2022.
- [26] ELEGIS. **Saiba quais são as principais fases do processo eleitoral**. Disponível em: <https://www.elegis.com.br/saiba-quais-sao-as-principais-fases-do-processo-eleitoral/>. Acesso em: 29 abr. 2022.
- [27] ESPANHA. **Guia de Referencia del DNIE com NFC**. Real Casa de la Moneda. Fábrica Nacional de Moneda y Timbre. 27 out. 2017. Disponível em: https://www.dnielectronico.es/PDFs/Guia_de_Referencia_DNIE_con_NFC.pdf. Acesso em: 15 abr. 2022.
- [28] ESPINER, Tom. **GCHQ pioneers on birth of public key crypto**. 26 out. 2010. Disponível em: <https://www.zdnet.com/article/gchq-pioneers-on-birth-of-public-key-crypto/>. Acesso em: 05 abr. 2022.

- [29] ESTÔNIA. **How to use your digital ID**. Disponível em: <https://learn.e-resident.gov.ee/hc/en-us/articles/360000624498-How-to-use-your-digital-ID>. Acesso em: 10 fev. 2022.
- [30] ESTÔNIA. **e-Identity**. e-Estonia. Disponível em: <https://e-estonia.com/solutions/e-identity/id-card/>. Acesso em: 15 abr. 2022.
- [31] HOLANDA. **DigiD**. Disponível em: <https://www.digid.nl/en/>. Acesso em: 15 abr. 2022.
- [32] IBM. **Public Key Certificates**. Disponível em: <https://www.ibm.com/docs/en/sdk-java-technology/8?topic=processes-public-key-certificates>. Acesso em: 15 abr. 2022.
- [33] International Telecommunications Union (ITU). **X.509: The Directory - Authentication framework**. 25 nov. 1988. Disponível em: <https://www.itu.int/rec/T-REC-X.509-198811-S>. Acesso em: 14 abr. 2022.
- [34] Internet Assigned Numbers Authority (IANA). **Transport Layer Security (TLS) Parameters**. 31 mar. 2022. Disponível em: <https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml>. Acesso em: 14 abr. 2022.
- [35] ITÁLIA. **The Electronic Identity Card (CIE)**. Carta di Identità Elettronica. Disponível em: <https://www.cartaidentita.interno.gov.it/en/home/>. Acesso em: 15 abr. 2022.
- [36] LEEUWEN, J. **Handbook of Theoretical Computer Science**. Elsevier, vol. 1, 1994.
- [37] MANEL, D.; MTIBAA, A.; RAMZI, H; RAOUF, O. **Hash function and Digital Signature based on elliptic curve**. 14th International Conference on Sciences and Techniques of Automatic Control & Computer Engineering, dez. 2013.
- [38] National Institute of Standards and Technology (NIST). **FIPS 186-4**. Disponível em: <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>. Acesso em: 17 fev. 2022.
- [39] NeritPolítica. **“Quais são as principais fases do processo eleitoral?”**. 16 jul. 2021. Disponível em: <https://neritpolitica.com.br/blog/quais-sao-as-principais-fases-do-processo-eleitoral>. Acesso em: 29 abr. 2022.

- [40] PARSOVS, Arnis. **Estonian electronic identity card and its security challenges**. Dissertationes Informaticae Universitatis Tartuensis, 3 mar. 2021. Disponível em: <https://dspace.ut.ee/handle/10062/71481>. Acesso em: 25 nov. 2021.
- [41] Receita Federal. **Perguntas e Respostas**. Disponível em: <https://receita.economia.gov.br/orientacao/tributaria/cadastros/cadastro-de-pessoas-fisicas-cpf/assuntos-relacionados/perguntas-e-respostas>. Acesso em: 29 jul. 2021.
- [42] REINO UNIDO. **James Ellis**. Government Communications Headquarters. 11 mar. 2019. Disponível em: <https://www.gchq.gov.uk/person/james-ellis>. Acesso em: 14 abr. 2022.
- [43] **RFC 2764**. A Framework for IP Based Virtual Private Networks. Disponível em: <https://datatracker.ietf.org/doc/html/rfc2764>. Acesso em: 28 abr. 2022.
- [44] **RFC 4251**. The Secure Shell (SSH) Protocol Architecture. Disponível em: <https://datatracker.ietf.org/doc/html/rfc4251>. Acesso em: 28 abr. 2022.
- [45] **RFC 6979**. Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA). Disponível em: <https://datatracker.ietf.org/doc/html/rfc6979>. Acesso em: 05 abr. 2022.
- [46] **RFC 8446**. The Transport Layer Security (TLS) Protocol Version 1.3. Disponível em: <https://datatracker.ietf.org/doc/html/rfc8446>. Acesso em: 05 abr. 2022.
- [47] **RFC 8551**. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0. Disponível em: <https://datatracker.ietf.org/doc/html/rfc8551>. Acesso em: 28 abr. 2022.
- [48] ROHR, Altieres. **Megavazamentos de dados expõem informações de 223 milhões de números de CPF: Dezenas de arquivos foram disponibilizados publicamente e colocados à venda por criminosos**. G1 - Economia. 25 jan. 2021. Disponível em: <https://g1.globo.com/economia/tecnologia/blog/altieres-rohr/post/2021/01/25/vazamentos-de-dados-expoem-informacoes-de-223-milhoes-de-numeros-de-cpf.ghtml>. Acesso em: 29 jul. 2021.

- [49] ROSA, Pedro Luiz Barros Palma da. **“Como funciona o sistema proporcional?”**. Escola Judiciária Eleitoral. Revista eletrônica EJE n. 5, ano 3. Disponível em: <https://www.tse.jus.br/o-tse/escola-judiciaria-eleitoral/publicacoes/revistas-da-eje/artigos/revista-eletronica-eje-n.-5-ano-3/como-funciona-o-sistema-proporcional>. Acesso em: 29 abr. 2022.
- [50] SENADO FEDERAL. **Glossário: Voto Majoritário**. Disponível em: <https://www12.senado.leg.br/noticias/glossario-legislativo/voto-majoritario>. Acesso em: 29 abr. 2022.
- [51] SERPRO. **DNI: a identidade unificada e digital do brasileiro**. 05 de junho de 2018. Disponível em: <https://www.serpro.gov.br/menu/noticias/noticias-2018/dni-a-identidade-unificada-e-digital-do-brasileiro>. Acesso em: 25 out. 2021.
- [52] SHOSTACK, Adam. **An Overview of SSL (version 2)**. Mai. 1995. Disponível em: <https://shostack.org/files/essays/ssl/>. Acesso em: 28 abr. 2022.
- [53] SMITH, R. E. **Elementary Information Security**. Jones & Bartlett Learning, 2016, ed. 2, p. 151.
- [54] VINHAS, Ana. **Em um ano, PF abre 931 inquéritos sobre fraude do auxílio: Desde o início do programa, em abril de 2020, foram realizadas 332 operações, 44 prisões e R\$1 milhão de bens apreendidos**. R7, [S. l.], 15 de maio de 2021. Disponível em: <https://noticias.r7.com/economia/em-um-ano-pf-abre-931-inqueritos-sobre-fraude-do-auxilio-15052021>. Acesso em: 14 jul. 2021.
- [55] VITORIO, Tamires. **Site brasileiro expôs 426 milhões de dados pessoais, diz empresa de segurança**. CNN, 22 de setembro de 2021. Disponível em: <https://www.cnnbrasil.com.br/business/site-brasileiro-expos-426-milhoes-de-dados-pessoais-diz-empresa-de-seguranca/>. Acesso em: 25 out. 2021.
- [56] WILSON, Stephen. **The Importance of PKI Today**. 2005. Disponível em: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.467.753&rep=rep1&type=pdf>. Acesso em: 10 fev. 2022.