PONTÍFICIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA DE ARTES
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SEGURANÇA EM MEIOS DE PAGAMENTO COM NFC

WALKER FREITAS DOS SANTOS

GOIÂNIA
2025

WALKER FREITAS DOS SANTOS

SEGURANÇA EM SISTEMAS DE PAGAMENTO COM NFC

Trabalho de Conclusão de Curso apresentado à Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para a obtenção do título de Bacharel em Ciência da Computação.

Orientador(a):
Prof.ª. Me Angélica da Silva Nunes

Banca examinadora:
Prof.º. Me Fernando Gonçalves Abadia
Prof.º. Me Rafael Leal Martins

GOIANIA
2025

WALKER FREITAS DOS SANTOS

SEGURANÇA EM SISTEMAS DE PAGAMENTO COM NFC

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Ciência da Computação, em _____/_____/_____.

_____

Orientador(a): Prof.ª Me Angélica da Silva Nunes

_____

Prof.º Me. Fernando Gonçalves Abadia

_____

Prof.º. Me. Rafael Leal Martins

GOIANIA

2025

DEDICATÓRIA

Agradeço a Deus, pela luz que iluminou o meu caminho. Pela força inabalável concedida nos momentos de maior desafio.

À minha amada família, meu porto seguro e a razão de todo o meu esforço. E principalmente à minha mãe, pelo amor incondicional, pelos incentivos constantes e por acreditarem no meu potencial.

Aos meus amigos, fonte de inspiração e que sempre me motivam a ser melhor e continuar crescendo a cada dia.

AGRADECIMENTOS

Primeiramente, agradeço a minha mãe, Ivonete, minha avó, Maria Rosália, aos meus padrinhos e tios, que sempre me suportaram e me auxiliaram em todas as etapas da minha vida

À professora Ma. Angélica da Silva Nunes, agradeço profundamente pela orientação, apoio e principalmente paciência, durante toda essa jornada

Aos meus amigos Samuel Spineli e Diego da Silva Ferreira, que mesmo a distância, mantiveram contato e me ajudaram a preservar minha sanidade durante todo o projeto.

Por fim, agradeço a todos que participaram e colaboraram para o sucesso desta jornada.

“E apliquei o meu coração a conhecer a sabedoria e a conhecer os desvarios e as loucuras, e vim a saber que também isto era aflição de espírito. Porque na muita sabedoria há muito enfado; e o que aumenta em conhecimento, aumenta em dor.”

Eclesiastes 1:17,18

RESUMO

O presente trabalho, intitulado "SEGURANÇA EM MEIOS DE PAGAMENTO COM NFC", analisa a aplicação da tecnologia Near Field Communication (NFC) em sistemas de pagamento, focando na segurança e na implementação em dispositivos Android. O estudo, baseado em revisão bibliográfica, abordou o funcionamento da NFC, a arquitetura de pagamentos por aproximação e os mecanismos de segurança empregados. O foco do estudo foi a arquitetura EMV (Eurocard, Mastercard e Visa) , explorando padrões de segurança como criptografia, tokenização e o uso de Elementos Seguros (*Secure Elements* - SE) para proteção e autenticação de transações. O desenvolvimento prático utilizou Android Studio e Kotlin. O foco foi o uso das classes nativas Android, como IsoDep e NfcA, para comunicação de baixo nível via mensagens APDU (Application Protocol Data Unit). Testes envolveram o envio de comandos APDU (SELECT DF, SELECT APPLICATION) a um smartcard para validar a comunicação e a interpretação das respostas. Os resultados demonstraram que o uso da documentação padrão (Android, EMV e ISO), sem a implementação de canais de segurança avançados (como Secure Messaging ou o protocolo Blinded Diffie-Hellman - BDH), impediu a obtenção de informações sensíveis do usuário, mas que é possível iniciar comunicação básica com uma *tag* NFC com suporte as classes IsoDep. Conclui-se que o acesso a dados críticos exige a ativação de mecanismos de segurança mais complexos e chaves de criptografia.

Palavras-Chave: NFC, EMVCo, Android, IsoDep, Segurança

ABSTRACT

This work, entitled "SECURITY IN PAYMENT METHODS WITH NFC," analyzes the application of Near Field Communication (NFC) technology in payment systems, focusing on security and implementation in Android devices. Based on a literature review, the study addressed the operation of NFC, the architecture of contactless payments, and the security mechanisms employed. The study focused on the EMV architecture (Eurocard, Mastercard, and Visa), exploring security patterns such as cryptography, tokenization, and the use of Secure Elements (SE) for transaction protection and authentication. Practical development utilized Android Studio and Kotlin. The focus was on the use of native Android classes, such as IsoDep and NfcA, for low-level communication via APDU (Application Protocol Data Unit) messages. Tests involved sending APDU commands (SELECT DF, SELECT APPLICATION) to a smartcard to validate communication and the interpretation of responses. The results demonstrated that using standard documentation (Android, EMV, and ISO), without implementing advanced security channels (such as Secure Messaging or the Blinded Diffie-Hellman - BDH protocol), prevented the acquisition of sensitive user information, but that it is possible to initiate basic communication with an NFC *tag* supporting IsoDep classes. It is concluded that accessing critical data requires the activation of more complex security mechanisms and encryption keys.

Keywords: NFC, EMVCo, Android, IsoDep, Security

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

AES-CTR	*Advanced Encryption Standard in Counter Mode,* Padrão de Criptografia Avançada em Modo Contador
AID	*Application Identification,* Identificador de Aplicação
AID	*Application Identifier*, Identificador da Aplicação
APDU	*Application Protocol Data Unit*, Unidade de dados de protocolo de aplicativo
ARAT	*Active Reader Active Tag,* Leitor Ativo de *tag* Ativa
ARPT	*Active Reader Passive Tag,* Leito Ativo de *tag* Passiva
BDH	*Blinded Diffie-Hellman*
CID	Confidencialidade, Integridade e Disponibilidade
C-RP	*Command-Response Pair,* Par Comando-Resposta
CVM	*Cardholder Verification Method,* Método de Verificação de Portador do Cartão
DF	*Dedicated File*
ECC	*Eliptic Curve Criptography,* Criptografia de Curva Elíptica
EMV	Eurocard, Mastercard e Visa
HCE	*Host-based Card Emulation*, Emulação de Cartão Baseada em *Host*
IDE	*Integrated Development Environment*, Ambiente de Desenvolvimento Integrado
IFF	Amigo ou Inimigo, *Identification Friend or Foe*
IO	*Input – Ouput*, Entrada – Saída
ISO	*International Organization for Standardization,* Organização Internacional de Normalização
JIS	*Japanese Industrial Standard,* Padrão Industrial Japonês
MITM	*Man-in-the-Middle,* Homem no Meio
MMVC	*Model–Model–View–Controller*, Modelo-Modelo-Visão-Controlador
NDEF	*NFC Data Exchange Format,* Formato NFC para troca de dados
NFC	*Near Field Comunication*, Comunicação em Campo Próximo
NFCIP	*Near Field Communication Interface and Protocol*
PCM	*Passive Communication Mode,* Modo de Comunicação Passiva

PIX	*Proprietary Application Identifier Extension*, Identificador de Aplicação Proprietária
POS	*Point of Sale,* Ponto de Venda
PRAT	*Passive Reader Active Tag,* Leitor Passivo de *tag* Ativa
RF	*Radio Frequency,* Rádio Frequência
RFID	*Radio-Frequency Identification,* Identificação por Radiofrequência
RID	*Registered Application Provider Identifier*, Identificador do Provedor de Aplicação Registrado
RSA	*Rivest-Shamir-Adleman*
SDK	*Software Development Kit*
SH	*Secure Messaging,* Comunicação Segura
UI	*User Interface*, *Interface* de Usuário

SUMÁRIO

1 INTRODUÇÃO

Segundo levantamento conduzido pelo *Near Field Communication Forum*, aproximadamente 85% dos consumidores, em nove países, utilizaram cartões de pagamento por aproximação ou carteiras digitais baseadas em *Near Field Communication*, Comunicação por Campo de Proximidade (NFC) entre os anos de 2020 e 2022. Esse índice representa um aumento de 30% no uso de cartões sem contato em relação ao ano de 2020. De acordo com o relatório, esse crescimento foi pela conveniência proporcionada pelas tecnologias sem contato e pelas preocupações sanitárias decorrentes da pandemia da COVID-19, que aceleraram a adoção de métodos de pagamento que evitam o contato físico. (EverythingRF, 2022)

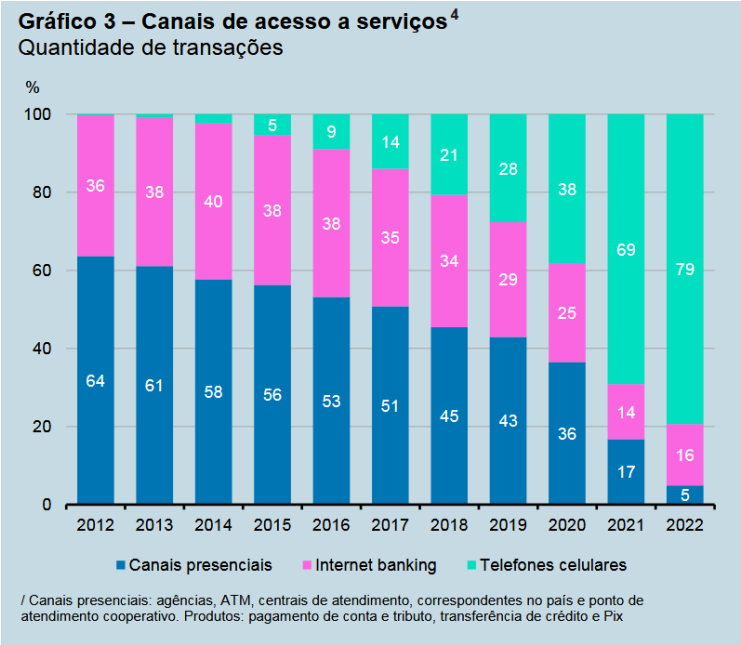
Entre a década de 2012 a 2022, o setor de pagamentos identificou um crescimento no uso de meios de pagamento digitais, motivado principalmente pela demanda por soluções ágeis e seguras. (Banco Central do Brasil, 2022)

Nesse contexto, a tecnologia NFC, consolidou-se como uma solução eficiente para transações de curto alcance. Sua popularidade é particularmente evidente em centros urbanos, caracterizados pela alta rotatividade de consumidores exige rapidez e praticidade. A crescente adoção de sistemas *contactless* é visível no crescimento de soluções como o "*Tap to Pay*", se tratando de uma iniciativa da empresa de pagamentos Visa, que permite pequenos comerciantes utilizem *smartphones* como terminais de pagamento. Esse *Model*o apresentou rápido crescimento, totalizando um aumento de 234% entre os países com maior adoção (Reino Unido, Brasil e USA) entre 2023 e 2024 (Businesswire, 2025).

Com o avanço da adoção de tecnologias, cresce também a necessidade de reforçar os mecanismos de segurança aplicados às comunicações NFC. Nesse cenário, o padrão *Eurocard, Mastercard and Visa* (EVM) *Contactless*, desenvolvido pelas empresas de mesmo nome do padrão, consolidando-se como um padrão de mercado ao ser usado por diversas empresas de transações financeiras. No entanto, conforme evidenciado em trabalhos como o de Akter (2021), ainda persistem vulnerabilidades estruturais no ecossistema NFC. Ataques sofisticados do tipo *Man-in-the-Middle,* Homem no Meio (MITM) continuam sendo tecnicamente viáveis, mesmo diante da aplicação de protocolos criptográficos robustos, o que evidencia a necessidade contínua de investigação e aprimoramento dos *Model*os de segurança adotados.

Segundo pesquisa de mercado realizada pelo Banco do Brasil (2022), observa-se um crescimento contínuo no uso de *smartphones* como plataforma para pagamentos digitais. O mercado de pagamentos via dispositivos móveis apresenta uma dominância significativa em relação a outras formas de pagamento. A Figura 1 ilustra esse crescimento anual.

Figura 1: Quantidade de transações por plataforma



Fonte: (Banco Central do Brasil, 2022)

Em ambientes móveis, soluções padrão para pagamentos digitais são, respectivamente, o Google Pay, para dispositivos Android, e o Apple Pay, para dispositivos da Apple. A plataforma Apple Pay contabiliza aproximadamente 747 milhões de usuários e movimenta cerca de 6 trilhões de dólares por ano (Revankar, 2025), consolidando-se como um agente relevante no setor de pagamentos móveis. Simultaneamente, observa-se um crescimento de 23% na adoção do Google Pay, o que evidencia a ampla utilização dessas tecnologias pela população em geral (Elad, 2025).

1.1 Justificativa

Diante da popularização da tecnologia NFC por consumidores, vendedores e a constante atenção a respeito da segurança dessa tecnologia emergente, surge a pergunta: qual o nível de suporte da plataforma Android para a tecnologia NFC de forma nativa e como padrões, como a EMVCo, influenciam nessas implementações?

1.2 Objetivo geral

Analisar as documentações e padrões relacionados à tecnologia NFC e implementar uma aplicação Android que permite operações de I/O a uma *tag*, utilizando as bibliotecas padrões oferecidas pelo sistema Android e em conformidade com as orientações EMV.

1.3 Objetivos específicos

- Identificar os principais protocolos de segurança utilizados no padrão EMV *Contactless*;
- Identificar as principais documentações e padrões relacionados a tecnologia NFC;
- Analisar os mecanismos de segurança disponibilizados pelas documentações oficiais NFC;
- Analisar a documentação oficinal do Android sobre a tecnologia NFC;
- Desenvolver uma aplicação para Android com suporte para comunicação NFC (IsoDep e NfcA) usando as bibliotecas oficiais do Android;
- Realizar testes de IO em uma *tag* NFC;
- Interpretar respostas enviadas por um dispositivo NFC;

1.4 Metodologia

O presente estudo, quanto a sua natureza, trata-se de um resumo de assunto, visto que irá condensar, sintetizar, analisar e discutir conhecimentos, informações e dados sobre o tema proposto, aplicando na prática o conhecimento abordado. (WAZILAWICK, 2014).

Em relação aos objetivos esta pesquisa classifica-se como explicativa, visto que o intuito é buscar um entendimento completo sobre o tema, discorrendo sobre o impacto, vantagens e em quais contextos *smartphones* diferem de *smartcards* em pagamentos via NFC. (WAZILAWICK, 2014).

No que tange aos procedimentos técnicos, caracteriza-se como uma pesquisa bibliográfica e experimental. Este procedimento é caracterizado pela manipulação de um aspecto da realidade pelo pesquisador. A pesquisa experimental implica ter uma ou mais variáveis experimentais que podem ser controladas, cuja medição poderá levar, possivelmente, à conclusão de que existe algum tipo de dependência com a variável experimental. (WAZILAWICK, 2014)

1.5 Estrutura da monografia

Este trabalho foi dividido em 6 capítulos.

O capítulo 1 é apresentado a introdução do tema do trabalho, com secções de objetivos gerais e específicos, procedimentos metodológicos.

O capítulo 2 apresenta conceitos básicos para entendimento do trabalho e a história da tecnologia NFC.

Capítulo 3 aborda as principais normas que regem a tecnologia NFC, abordando também padrões e métodos de segurança durante a comunicação entre dispositivos.

O capítulo 4 apresenta conceitos básicos de desenvolvimento Android e as classes responsáveis pela comunicação entre um dispositivo móvel e uma *tag* NFC

O capítulo 5 descreve os procedimentos metodológicos utilizados.

Capítulo 6 apresenta a conclusão, com a importância deste trabalho para melhor entendimento da comunicação, segurança e implementação da tecnologia NFC em dispositivos Android.

2 CONCEITOS

A tecnologia NFC, permite a troca de dados sem fio entre dispositivos próximos, sendo amplamente utilizada em sistemas de pagamento, autenticação e identificação. Dito isso, a segurança se torna um pilar nas implementações e usos de tecnologias NFC. Esse capítulo aborda os conceitos fundamentais de segurança para a tecnologia NFC, além do funcionamento básico de sistemas de *Radio Frequency*, Rádio Frequência (RF).

2.1 Segurança

Guttman e Roback (1995, p. 2) definem segurança da informação como:

> "The protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in order to ensure confidentiality, integrity, and availability."

Traduzindo:

> "A proteção das informações e dos sistemas de informação contra acessos, usos, divulgações, interrupções, modificações ou destruições não autorizadas, com o objetivo de assegurar a confidencialidade, a integridade e a disponibilidade".

2.1.1 Pilares da segurança

Segundo Stallings e Brown (2014), um dos objetivos da segurança da informação é assegurar a tríade Confidencialidade, Integridade e Disponibilidade (CID). Esses três pilares formam a base conceitual das políticas de segurança da informação:

- Confidencialidade: garante que informações sensíveis sejam acessadas apenas por indivíduos autorizados;
- Integridade: assegura que os dados e sistemas permaneçam corretos e completos, permitindo modificações apenas por fontes autorizadas;
- Disponibilidade: refere-se à garantia de que os sistemas e dados estejam acessíveis quando necessários pelos usuários autorizados.

2.1.2 Criptografia

A criptografia é uma área da matemática que estuda métodos de conversão de dados chamados de sistemas criptográficos, de forma que um texto claro, seja transformado em um texto cifrado, com o conteúdo embaralhado e sem nenhuma semelhança com texto original. O processo de converter o texto claro em texto cifrado é conhecido como cifração e o processo inverso é conhecido como decifração. (Stalings, 2014)

2.1.2.1 Algoritmos de criptografia assimétrica

Alguns algoritmos de criptografia assimétrica são (Stallings, 2014):

- RSA: o algoritmo Rivest-Shamir-Adleman (RSA) é uma técnica de encriptação assimétrica, mundialmente utilizada para criptografia de chave pública e privada;
- Diffie-Hellman: o algoritmo Diffie-Hellman é usado para realizar a troca de uma chave qualquer, de forma segura por um meio muitas vezes inseguro. Sistemas NFC destinados a transações financeiras utilizam desse algoritmo para o compartilhamento de chaves geradoras usadas para encriptar e autenticar transações;
- Curvas Elípticas: o algoritmo de curvas elípticas, *Eliptic Curve Cryptography*, Criptografia de Curvas Elípticas (ECC) tem como principal benefício a aparente simplicidade e eficiência do sistema, graças ao tamanho de chave menor, ao mesmo tempo que oferece o mesmo nível de segurança que o RSA, tornando-o muito usado em aplicações NFC, no qual o tamanho de chave deve ser preferencialmente pequeno, visto que o tamanho limitado de armazenamento e o tempo de transmissão de dados entre dispositivos.

2.1.3 Hash

O *hash* é uma função matemática que recebe uma mensagem *MM* de tamanho variável como entrada e gera um texto *XX* de tamanho fixo e de aparência aleatória, com o objetivo de assegurar a integridade da mensagem original. Qualquer alteração em *MM* resulta em um valor de *hash* diferente. No contexto da segurança da

informação, uma função *hash* deve ser computacionalmente inviável tanto para encontrar um objeto de dados que corresponda a um valor de *hash* previamente especificado (propriedade de mão única) quanto para identificar dois objetos distintos que resultem no mesmo valor de *hash* (propriedade de resistência a colisões). (Stallings, 2014).

2.2 RFID/NFC

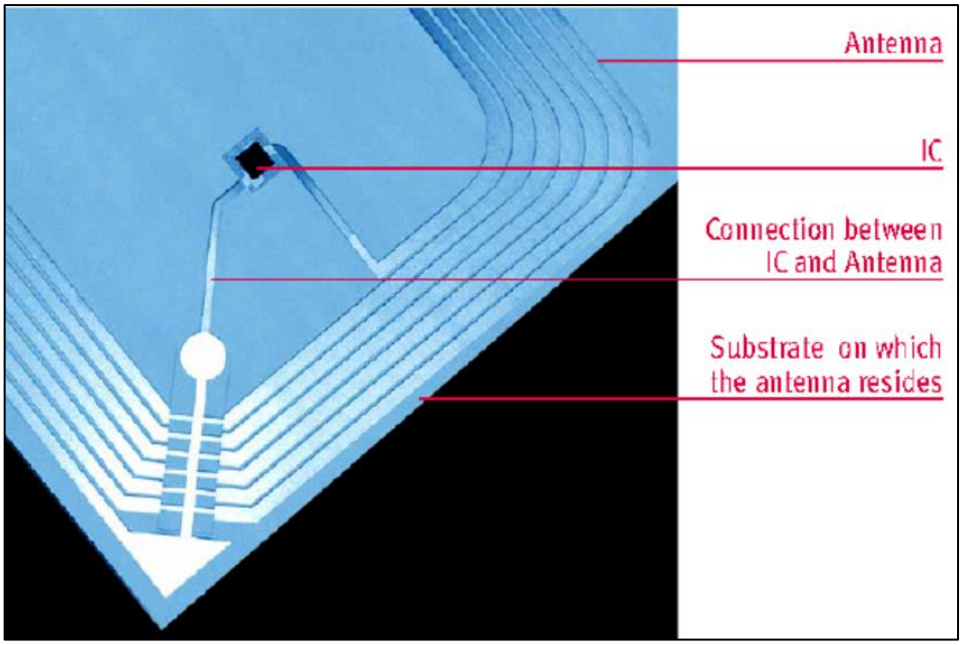
O NFC é um conjunto de protocolos que permite a comunicação sem fio entre dispositivos eletrônicos até uma certa distância, dependendo da finalidade do dispositivo. O NFC tem como base a tecnologia de Identificação por *Radio-Frequency Identification*, Rádio Frequência, (RFID), que permite que, com o uso de bobinas elétricas, dois dispositivos eletrônicos comuniquem estre si, de forma *wireless*, em alguns casos, mesmo que um dos dispositivos esteja descarregado. (BasuMallick, 2022)

2.2.1 Funcionamento e campos de rádio frequência

A tecnologia RFID tem como base a evolução de conceitos desenvolvidos a partir da Segunda Guerra Mundial, com a criação de sistemas de *transponders Identification Friend or Foe*, Amigo ou Inimigo (IFF), que permitiam a diferenciação automática entre aeronaves aliadas e inimigas, utilizando *transponders* que respondiam a sinais de rádio emitidos a partir de estações de solo. Partindo desse conceito, em 1969, Mario Cardullo formalizou a ideia de um *transponder* passivo com memória regravável, utilizando sinais de rádio frequência como fonte de energia e comunicação. Dessa forma, a primeira patente de RFID nasceu. (Violino, 2005)

A tecnologia RFID usa *tags* ou *tokens* para identificação e armazenamento de dados. Um rádio transmissor, chamado de *interrogators* ou *reader*, envia um sinal para uma *tag* que, por sua vez, deve retornar uma resposta ao *reader*, de forma ativa ou passiva. Uma *tag* RFID é composta de três componentes: um *chip* RFID, usado para receber e guardar informações; uma antena, que recebe sinais de um emissor; e um substrato, que mantem os componentes juntos. A Figura 2 apresenta um diagrama de um *Smartcard* RFID. (PARDAL, 2010)

Figura 2: Diagrama de uma *tag* RFID



Fonte: (PARDAL, 2010)

Leitores de RFID variam conforme sua aplicação, especialmente em relação à frequência de sinais emitidos. Quanto à tipologia, classificam-se em três *Model*os principais:

- *Passive Reader Active Tag* (PRAT): o leitor atua de forma passiva, apenas recebendo sinais de *tags* ativas, que possuem bateria própria. Essa configuração permite alcances de até 600 metros;
- *Active Reader Passive Tag* (ARPT): o leitor é ativo e emite sinais aos quais etiquetas passivas respondem. É uma solução eficiente e de baixo custo para ambientes controlados;
- *Active Reader Active Tag* (ARAT): Ambos leitor e etiqueta são ativos. Essa configuração oferece maior alcance e desempenho, sendo ideal para aplicações críticas que demandam comunicação contínua. (Phillips, 2017)

2.2.1.1 Smartcard

Um *smartcard* pode ser definido como um circuito integrado em um cartão, contendo componentes para transmitir, guardar e processar dados. Os dados contidos no cartão podem ser transmitidos usando tanto contato físico com o *chip* integrado no cartão (*contact card*) ou sem contato algum (*contactless*). (Rankl; Effing, 2004)

Especificações físicas, protocolos, frequências e outras características de cartões *contact* e *contactless* são governadas pelas ISO/IEC 7816 e ISO/IEC 14443 respectivamente, além de poderem seguir também outras especificações, como a NFC-Forum e EMV.

2.2.2 Modos de operação

Uma *tag* NFC pode adotar diferentes modos de operação indicados para diferentes casos de uso. Para todos esses casos, define-se como modo ativo um dispositivo NFC que gera um campo RF e inicia a comunicação, e passivo como um dispositivo que não gera um campo RF e necessita de um campo RF de outro dispositivo para se energizar e executar operações. (ISO/IEC 18092, 2023).

Quanto a um *reader*, três modos de operação podem ser adotados, *Read and Write*, *Peer to peer* e *Card Emulation*, Figura 3 ilustra os diferentes modos e os protocolos utilizados.

2.2.2.1 Reader/writer modules

Os *Reader/writer modules* são também conhecidos como *Passive Communication Mode*, Modo Passivo de Comunicação (PCM), um dispositivo NFC ativo atua como um *reader* ou *writer* que pode ler e escrever informações em outros dispositivos NFC, sendo eles *Tags* NFC ou *contactless cards*. (Thorat; Laulkar, 2025).

O procolo de comunicação padão é o *NFC Data Exchange Format,* Formato de Intercâmbio de Dados NFC (NDEF) definido pela *NFC Forum*, podendo variar a depender do tipo de *tag* trabalhada, com certas *tags* (a exemplo da classe *Type* 3 que segue o *Japanese Industrial Standard,* Padrão Industrial Japonês (JIS)) podendo usar padrões proprietários.

2.2.2.2 Peer to peer mode

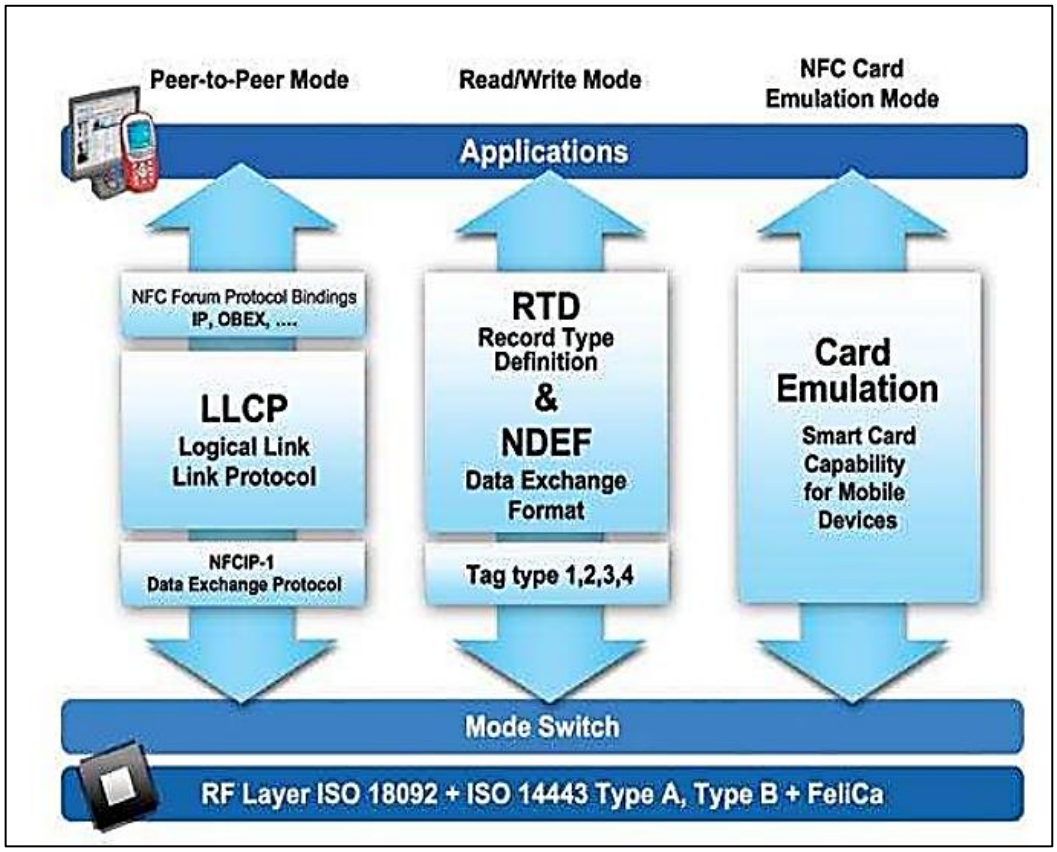
Modo *Peer to Peer*, também conhecido como modo ativo, dois dispositivos NFC trocam informações entre si, permitindo que cada dispositivo execute ações de leitura

e escrita no outro. Nesse modo, um dispositivo atua como um *initiator* e outro como um *target*. Dessa forma mensagens são enviadas de um iniciador, que gera o próprio campo RF, para um *target* e os dispositivos podem trocar esses papeis a qualquer momento, permitindo comunicação bidirecional. (BasuMallick, 2022)

2.2.2.3 Card emulation

No modo *Card emulation* o dispositivo emula um *Contactless Smart Card* seguindo as especificações básicas da *International Organization for Standardization,* Organização Internacional de Normalização, (ISO). Dessa forma um dispositivo pode se comunicar com qualquer outro dispositivo NFC que segue as normas padrões da NFC Fórum ou ISOs 7816 ,14443, e 18092. Normalmente esse modo é usado para a realização de transações financeiras e outros sistemas de autenticação (Thorat; Laulkar, 2025).

Figura 3: Modos de operação NFC



Fonte (Thorat; Laulkar, 2025)

3 SEGURANÇA EM MEIOS DE PAGAMENTO VIA NFC

Neste capítulo, são apresentados os principais conceitos que sustentam o funcionamento e a aplicação da tecnologia NFC com ênfase em meios de pagamento, abordando padrões e normas internacionais, definições básicas, bem como etapas de comunicação.

3.1 Normas

As normas relacionadas à tecnologia NFC são diversas e variam conforme o caso de uso. As normas ISO/IEC, que incluem especificações como a série *Near Field Communication Interface and Protocol* 1 e 2*,* (NFCIP) que estabelecem aspectos básicos da tecnologia, incluindo características físicas de leitores e etiquetas (*tags*). Complementarmente, o NFC *Forum* é responsável pela definição das especificações gerais que abrangem as diferentes classes de dispositivos NFC. No contexto de transações financeiras, a EMVCo, em colaboração com o NFC *Forum* e a ISO/IEC, desenvolve padrões específicos voltados para a interoperabilidade e segurança de pagamentos realizados por meio de dispositivos NFC e *smartcards*.

3.1.1 Normas ISO

Diversas normas ISO/IEC existem para definir padrões para tecnologias relacionadas a *smartcards* e NFC. A ISO/IEC 7816 trata das especificações para *contact smartcards* abordando aspectos como estrutura elétrica, protocolos de transmissão e formato de dados.

As ISO/IEC 18092 e ISO/IEC 21481, respectivamente também conhecidas como NFCIP-1 e NFCIP-2. A ISO/IEC 18092 definem parâmetros técnicos para a comunicação NFC entre dispositivos, como *smartphones* e leitores, estabelecendo velocidades de transmissão, modos de operação e requisitos de interoperabilidade. A ISO/IEC 21481 complementa a ISO 18092, especificando modos de comunicação para coexistência entre diferentes tecnologias NFC e RFID. (ISO/IEC 18092, 2023; ISO/IEC 18092, 2023)

3.1.2 NFC fórum

O NFC Forum é uma associação sem fins lucrativos fundada por *Nokia Corporation*, *Royal Philips Electronics* e *Sony Corporation*. Seu objetivo principal é promover a implementação e padronização de tecnologias NFC, visando assegurar a interoperabilidade entre dispositivos e serviços que necessitam dessa tecnologia. (Secure Tech Alliance, 2008)

3.1.3 EMV

O EMV é uma série de especificações voltadas a *smartcards* usados como forma de pagamento, baseado nas diversas especificações ISO e NFC *Forum*, com sua primeira publicação em 1996. (WARD, 2006).

Para garantir a manutenção e evolução coordenada do padrão EMV, Europay, MasterCard e Visa criaram a EMVCo em 1999, transferindo à nova entidade a responsabilidade pelo gerenciamento técnico e padronização global da tecnologia, adicionando novos membros e gerenciando a especificação EMV e ofertando as devidas especificações para o padrão, na forma das especificações EMV *Books*. (EMVCo, 2014)

3.1.3.1 Tipos de especificações

A EMVCo disponibiliza um conjunto especificações técnicas que visam definir protocolos e outros padrões que visam a interoperabilidade e a segurança em transações de pagamento com *smartcards* e dispositivos móveis. Essas especificações são organizadas em séries de documentos conhecidos como "*Books*", que abordam tanto operações com *contact smartcards* quanto transações *contactless*, (que incluem a tecnologia NFC).

Os livros A e E das especificações da EMVCo tratam dos requisitos para transações de pagamento sem contato. Contendo especificações de comandos, requisitos do sistema, tipos de criptografia, processos etc. (EMVCo, [*S. d*])

3.1.3.2 *Kernels,* sistemas POS, *entrypoint* e outras definições

A referência *EMV Contactless Specifications for Payment Systems Book A,* EMVCo (2023), define os principais atores de todo o processo de transação de dados entre dispositivos NFC. Os componentes são:

- Cartão (*Card*): na especificação EMVCo, o termo "cartão" refere-se a qualquer *token* com suporte a transações de pagamento por aproximação, independentemente de seu formato físico. Isto inclui cartões com *chip* para pagamentos *contactless*, chaveiros, *smartphones* ou outros dispositivos;
- Transação (*Transaction*): a especificação EMVCo define uma transação como uma troca de informações através de uma *interface contactless*, usando um cartão compatível. O início da transação pode ocorrer de duas formas: por ação de um comerciante, como a entrada do valor da compra, ou automaticamente, pela detecção de um cartão ao entrar no campo RF de leitura do terminal. A transação se estende até a comunicação final, enviada ao portador do cartão. Toda comunicação adicional com o cartão após será considerada parte da transação;
- POS *System*: o sistema *Point of Sale*, Ponto de venda (*POS*) é um dispositivo que se comunica com cartões *contactless*, processa operações *contactless,* podendo oferecer suporte a outros tipos de métodos de pagamento, usando, por exemplo, cartão magnético ou cartões por contato. É comum referenciar o sistema POS como apenas “leitor”;
- *Entry Point*: o *Entry Point* é um *software*, contido em um dispositivo POS, responsável pela pré-processamento da comunicação, descoberta e seleção de uma aplicação *contactless* compatível entre o cartão e o terminal, ativação do *kernel* apropriado e tratamento dos resultados (*outcomes*) retornados pelo *kernel;*
- *Kernel*: o *kernel* é o *software* no sistema POS que realiza o processamento da transação *contactless*, após ser selecionado pelo *Entry Point*. A seleção do *kernel* considera as características da transação, as aplicações suportadas por cartão e leitor, e o protocolos de controle e segurança. Múltiplos *kernel*s existem para atender as diferentes necessidades do mercado e são documentados na série de livros intitulados *Book C–x Kernel*

Specification, numerados de 1 a 8, onde cada numeração representa um tipo de *kernel* diferente;

- *Outcome*: o *Outcome* é a principal instrução fornecida pelo *kernel* sobre como o processamento de uma transação deve ser feito. Um *Outcome* é acompanhado por um conjunto de parâmetros que podem personalizar a experiência do usuário, através, por exemplo, de mensagens que podem ser exibidas ao longo de alguma transação;
- *Reader* (Leitor): o leitor é uma parte do terminal POS que provê a *interface* de comunicação para o cartão. (EMVCo, 2023);
- *Application:* a aplicação é uma série de estruturas, elementos de dados e módulos de programas necessários para realizar uma funcionalidade específica. (ISO 7816-4, 2020);
- *Application Identifier* (AID): o Identificador de aplicação é um código padronizado pela ISO 7816-4 que identifica uma *application*;
- *Cardholder Verification Method* (CVM): Um método de verificação que prova identidade do titular de um cartão, podendo ser *online*, *offline*, biométrico etc. (EMVCo, 2025)

3.2 Segurança durante o processo de comunicação

Os sistemas NFC passam por diversas etapas de segurança durante o processo de comunicação, iniciando através do processo de trocas de chaves com o algoritmo *Blinded Diffie-Hellman* (BDH), etapas de comunicação local, que permitem que o cartão usado passe por um processo de autenticação para garantir que o cartão é genuíno.

3.2.1 Canal de segurança

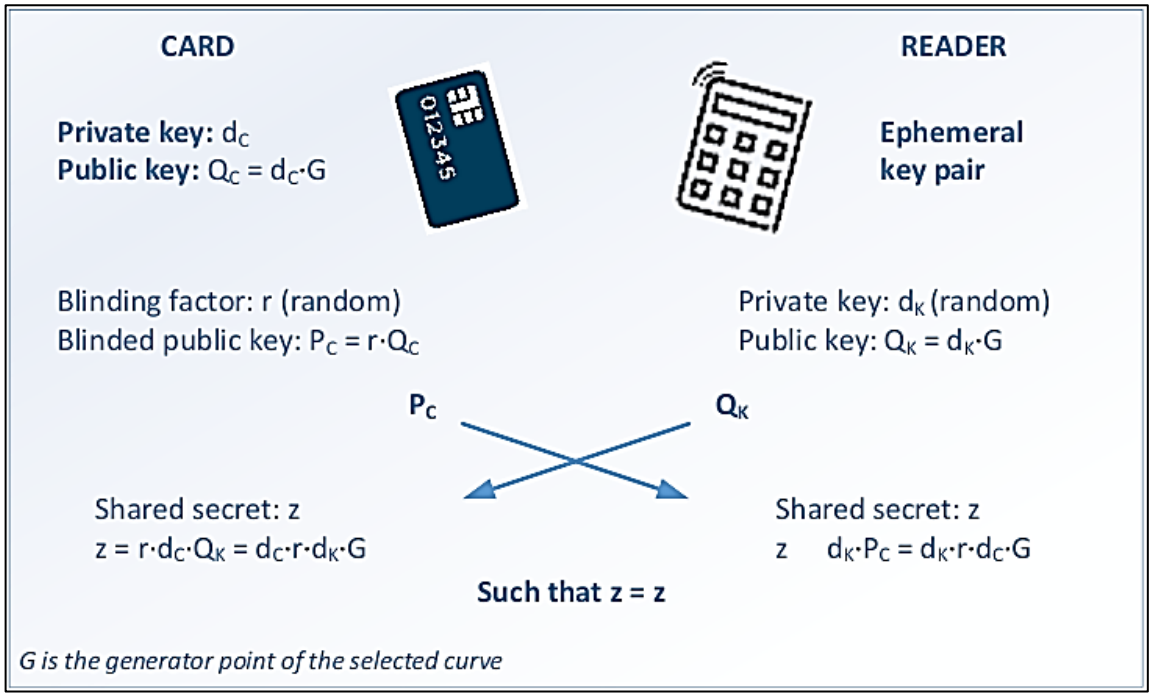
Durante uma transação um canal seguro entre o POS e o cartão *contactless* deve ser estabelecida. Esse canal garante privacidade, segurança durante o processo de armazenamento de dados e autenticação local. Para isso, o algoritmo BDH é necessário. Ambas as chaves contidas no cartão são emitidas apenas pela entidade certificadora, durante a emissão do cartão.

Durante o processo, um par de chaves ECC é gerada para cada transação. Como a chave pública de um cartão é fixa, esta necessita ser anonimizada para cada transação, isso é possível ao multiplicar a chave pública por um inteiro aleatório, chamado de *blinding factor* r.

Uma versão simplificada do processo BDH é descrito na Figura 4. Onde:

- Chave privada do cartão d_c.
- Chave pública do cartão Q_c.
- Número aleatório r.
- Chave pública ofuscada P_c, obtida após o cálculo $P_c = r. Q_c$.
- Chave temporária privada do leitor d_k.
- Chave temporária do leitor Q_k.
- Ponto gerador da curva elíptica G.
- Segredo compartilhado z definido pela entidade emissora.

Figura 4: Acordo BHD

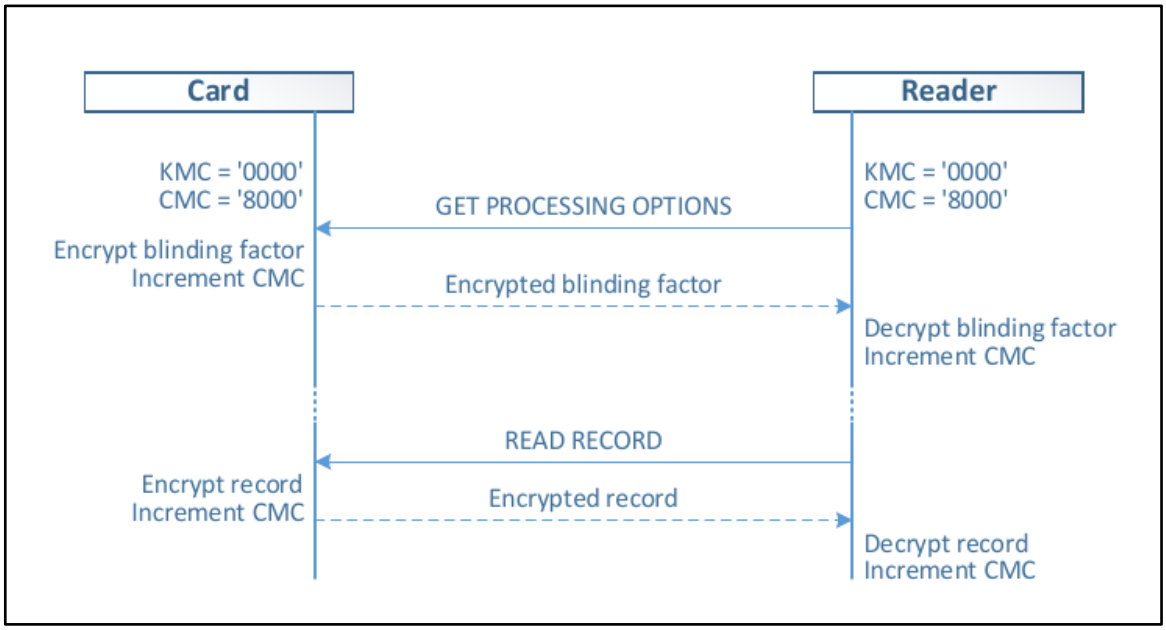


Fonte: (EMVCO, *Book E*, 2023)

Após estabelecido um canal de segurança, dados sensíveis retornados pelo cartão, são encriptados usando uma modificação do algoritmo AES *in counter mode*, Padrão de Criptografia Avançada em Modo Contador, (AES-CTR). Figura 5 exemplifica o processo de criptografia, onde um *Reader* envia um comando APDU

considerado sensível (algum dado que possa ser usado para identificar um cartão), com a *tag* retornando os dados criptografados. Dados considerados sensíveis são definidos pelo EMVco *Book E*, sendo alguns deles o *Application Primary Account Number,* Número de Conta Primário da Aplicação *(PAN),* Certificado de Chave Pública do Cartão e a Chave Pública ECC do Cartão.

Figura 5: Encriptação de dados



Fonte: (EMVCO, *Book E*, 2023)

3.2.2 Etapa de autenticação local e remota

Dois processos de autenticação são realizados durante a etapa de pré-processamento de informações e ambos são usados para provar a genuinidade do cartão usado e autenticar a transação de dados, a diferença entre os dois métodos está na forma em que a autenticação é feita.

A autenticação local visa autenticar o cartão com um banco de dados local, armazenado em uma memória interna do POS, através da comparação entre a chave pública da entidade Emissora, e o certificado ECC do cartão, obtido após o acordo BHD.

A autenticação remota visa, remotamente, autenticar o cartão e a transação executada usando um método de validação equivalente a autenticação local, mas enviando requisições a um servidor do provedor do cartão. Normalmente as duas formas de autenticação são feitas durante todo o processo de transação financeira.

3.3 Etapas de comunicação

Etapas de comunicação podem ser ignoradas dependendo do tipo de cartão, tipo de terminal POS ou algum caso específico. Cada etapa é identificada por meio de *starts*, inícios.

3.3.1 Start A – Pré-processamento

O *Pre-Processing* é a etapa inicial do *Entry Point* e do processo de transação. Nessa etapa, o dispositivo avalia configurações específicas do cartão pela combinação de AID e *Kernel ID*. Para cada combinação, são analisados dados como limites de valor (limite *contactless*, *floor limit* etc.), necessidade de CVM, capacidade *online/offline*, dentre outras funcionalidades definidas no *EMV Book A*.

3.3.2 Start B - Ativação de protocolo

O *Start* B inicia o processo de descoberta e comunicação entre o cartão e o sistema POS. Nessa etapa, mensagens são enviadas para a interface de usuário, o campo de leitura é energizado e o sistema realiza uma procura por dispositivos compatíveis próximos. Nessa etapa ocorre tratamento de colisão de múltiplos cartões, e restauração de dados previamente armazenados em caso de reinício do sistema POS.

3.3.3 Start C – Seleção de combinação

A etapa de seleção de combinação, *Combination Selection* seleciona a combinação AID e *Kernel* ID presentes no cartão e um *kernel* EMV suportado pelo terminal. A combinação definida determina qual estrutura de arquivos e dados é usada na transação e qual lógica de processamento o terminal deve seguir, com base na definição da aplicação EMV correspondente.

3.3.4 Start D – Ativação de kernel

A etapa de *Kernel Activation* consiste na transferência do controle do processo pelo *Entry Point* para o *kernel* associado à combinação selecionada. A partir desse ponto, o *kernel* passa a ser responsável pelo processamento de todos os comandos e respostas da transação. A imagem 4 representa esse processo.

3.3.5 Processamento de outcome

Ao final de cada processamento, todo *Kernel* retorna um *Outcome* com determinados parâmetros. Algum deles podem reiniciar a etapa de processamento a partir de algum *start* específico, enquanto outros *outcomes* finalizam a operação. O processo detalhado da operação de *outcome* é definido na especificação *Book A.*

4 CONCEITOS BÁSICOS DE DESENVOLVIMENTO ANDROID COM KOTLIN

Os seguintes capítulos comentam sobre o processo de desenvolvimento com *Kotlin* e Android

4.1 Desenvolvimento NFC em dispositivos Android,

Sistemas Android possuem nativamente ferramentas de suporte para comunicação entre dispositivos NFC. A forma mais simples de comunicação é através de mensagens formatadas para comunicação NFC ou NDEF. As mensagens NDEF são padronizadas pelo NFC Fórum e permite a comunicação entre diferentes dispositivos NFC que seguem os padrões NFC Fórum. No entanto, o padrão NDEF não é aplicado em dispositivos NFC destinados a sistemas de pagamento. Para isso, o EMVCo integra os padrões ISO (com as definições das mensagens desses padrões sendo definida na ISO 7816-4) com os próprios padrões.

4.1.1 Classes ISODep e MiFare

O Android disponibiliza outras formas de comunicação entre dispositivos NFC que não necessariamente implementam os padrões da NFC Foundation, permitindo o envio de comandos e respostas (na forma de *bytes* em hexadecimal) entre esses dispositivos. Para comunicação usando os protocolos ISO, o ambiente Android disponibiliza as classes NfcA, NfcB, NfcV e IsoDep. Outros pacotes que oferecem suporte adicional a outras especificações incluem: NfcF para comunicação entre dispositivos JIS; MifareClassic e MifareUltralight para comunicação entre dispositivos com *chip MiFare* através da classe android.nfc.tech.

4.1.2 Comunicação entre dispositivos usando IsoDep

A classe IsoDep (ISO-Data Exchange Protocol) é destinada para dispositivos que seguem o protocolo ISO/IEC 14443-4 e o protocolo de smartcard ISO/IEC 7816-4. O IsoDep atua como a interface de alto nível para a troca de dados em APDUs. Para que o corra a comunicação entre um dispositivo Android e uma *tag* NFC ´é necessário, primeiramente, que o dispositivo Android tenha suporte a tecnologia NFC

e que a funcionalidade esteja ativada no sistema. A partir do momento que o dispositivo do usuário é desbloqueado o sistema Android começa a procurar por *tags* compatíveis.

No momento que uma *tag* é detectada o sistema Android emite um "*Intent*" que a aplicação escuta e inicia o processo de leitura básica do dispositivo NFC:

Figura 6 Código para detecção de *tag* NFC

```
58 override fun onNewIntent(intent: Intent) {
59    super.onNewIntent(intent)
60    try {
61        if (intent != null && (intent.action ==
NfcAdapter.ACTION_TAG_DISCOVERED || intent.action ==
NfcAdapter.ACTION_TECH_DISCOVERED)) {
62            val tag: Tag? = if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.TIRAMISU) {
63                intent.getParcelableExtra(NfcAdapter.EXTRA_TAG,
Tag::class.java)
64            } else {
65                @Suppress("DEPRECATION")
66                intent.getParcelableExtra(NfcAdapter.EXTRA_TAG)
67            }
68            tag ?.let { nfcReaderViewModel.onTagScanned(it) } ?:
{Utils.w("Tag null");}
69        }else{
70            Utils.w("Intent null ou Tech não suportada");
71        }
72    }catch (e: IOException){Utils.e("Error during intent reading", e)}
73}
```

Fonte: (Elaborado pelo autor, 2025)

Ao detectar uma nova *tag*, o sistema cria um objeto *tag* que guarda um identificador temporário de uma *tag* específica. Esse identificador pode ou não ser gerado todas as vezes que a *tag* é escaneada. A Figura 6 representa o código usado para obter o *intent* de uma *tag* NFC, especificamente na linha 68, onde a função $onTagScanned(it)$ cria uma cópia do objeto Tag para ser usada na classe customizada. A Figura 7 mostra como as classes IsoDep e NfcA são usadas para

obter dados da *tag*, esses dados são mostrados pelo aplicativo na aba de “*Tag Info”* (Figura 8).

Figura 7: Obtendo metadados básicos da *tag*

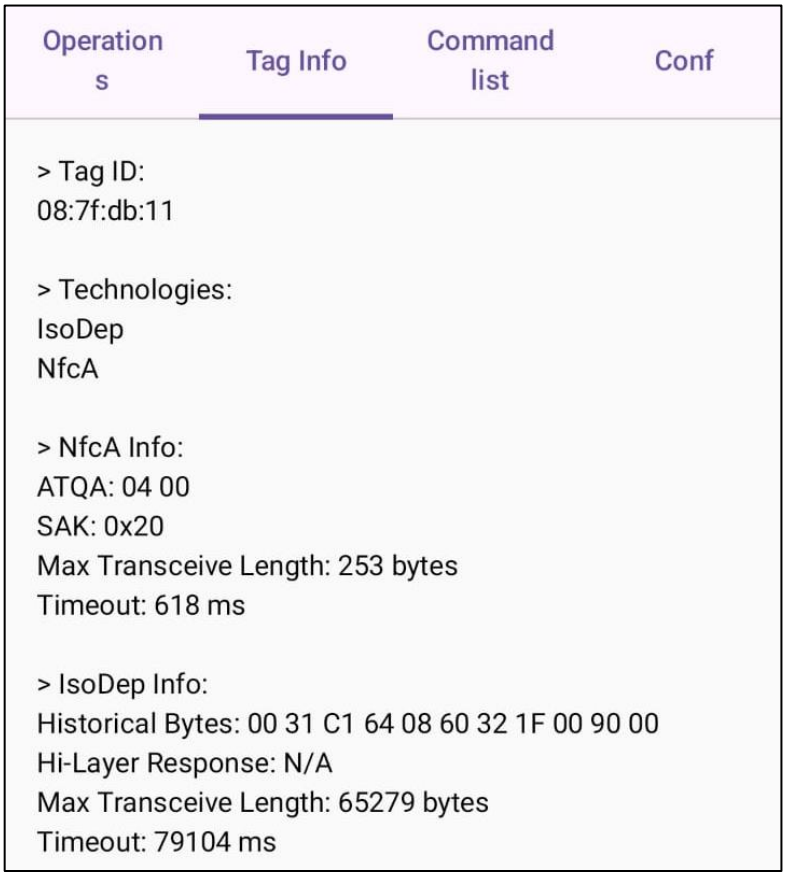
```
tagNfcA = NfcA.get(tag)
tagIsoDep = IsoDep.get(tag)

val atqaBytes = tagNfcA?.atqa
val sakShort = tagNfcA?.sak
val maxTransceiveLength = tagNfcA?.maxTransceiveLength
val timeout = tagNfcA?.timeout

val historicalBytes = tagIsoDep?.historicalBytes
val hiLayerResponse = tagIsoDep?.hiLayerResponse
val localMaxTransceiveLength = tagIsoDep?.maxTransceiveLength
val localTimeout = tagIsoDep?.timeout
```

Fonte: (Elaborado pelo autor, 2025)

Figura 8: Tela de informações básicas da *tag*



Fonte: (Elaborado pelo autor, 2025)

4.1.3 Comandos APDU

A comunicação entre o dispositivo e a *tag* NFC corre através do *Application Protocol Data Unit*, Unidade de dados de protocolo de aplicativo (APDU) que pode ser na forma de um comando enviado ao dispositivo NFC ou na forma de uma resposta. Um comando que espera uma resposta de um dispositivo é chamado de *Command-Response Pair,* Par Comando-Resposta (C-RP).

Usando a classe $IsoDep$, com a função $transceive()$ é possível enviar *bytes* de dados para uma *tag* NFC. Esses *bytes* em formato hexadecimal seguem o padrão APDU definido pela ISO 7816-3 e cada comando APDU segue certos padrões que podem mudar dependendo da tecnologia e *Kernel* usado.

Um comando APDU definido pela ISO 7816 consiste em um *Header* e um *body* que pode ser opcional.

Figura 9 Campos de um comando APDU padrão

Command header	Command body
CLA INS P1 P2	[L_c field] [Data field] [L_e field]

Fonte (ISO 7816-3, 2006)

Uma resposta de um comando APDU consiste em um *body* de tamanho variável e uma parte de *bytes* denominados SW1 e SW2 que contém o *status* da resposta.

Figura 10 Campos de uma resposta APDU padrão

Response body	Response trailer
[Data field]	SW1 SW2

Fonte (ISO 7816-3, 2006)

Os valores dos campos de comando e resposta são definidos pela ISO 7816-4, mas também podem ser definidos por outros padrões internacionais (como a EMVCo, *NFC Foundation*) ou por definições mais regionais.

4.1.3.1 Componentes de um comando e resposta APDU

Um comando APDU consiste em uma *string* de n *byte*s. O campo *Header* é composto dos primeiros quatro *bytes* do comando, onde o *body* consiste dos *bytes* restantes. A resposta pode conter n *byte*s, com os últimos quatro *bytes* sendo um código de status da operação.

Quadro 1 contém uma descrição básica dos campos de uma mensagem APDU.

Quadro 1: Campos de um comando APDU

Campo	Descrição	Tamanho em *byte*s	Direção
Campo de *Header*	Classe: CLA	1	Em direção a *Tag*
	Instrução*: INS*	1	
	Parâmetros: P1 e P2	2	
Campo de *body*	Campo L_c: Número de *bytes* do campo de dados	0, 1 ou 3	
	Campo de dados	L_c	
	Campo L_e: Quantidade de *bytes* esperados da resposta	0, 1, 2 ou 3	
Campo de resposta	Resposta do comando APDU	0 ou L_c	Em direção ao leitor
Status de resposta	*Bytes* de status	2	

Fonte: (ISO 7816-3, 2006)

Outras definições não necessárias para a continuidade deste trabalho foram omitidas. Cada um dos campos já citados é mais bem explicado na etapa de exemplos.

4.1.4 Identificador de aplicação (AID) de uma tag NFC

Um *Application Identifier*, Identificador da Aplicação (AID) é um identificador único utilizado em cartões inteligentes conforme a ISO/IEC 7816. Ele serve para selecionar e distinguir aplicações específicas que residem no chip como aplicações de pagamento EMV, programas de fidelidade ou aplicações proprietárias. Um AID é composto por um Identificador do Provedor de Aplicativos Registrado, *Registered*

Application Provider Identifier, Identificação do Provedor da Aplicação (RID), definido pela ISO 7816-5, seguido de uma Extensão de Identificador de Aplicativo Proprietário, *Proprietary Application Identifier Extension*, Extensão de Identificador de Aplicativo Proprietário (PIX), formando um identificador globalmente único que permite ao terminal escolher qual aplicação executar no cartão.

4.1.5 Exemplos de comandos e retornos APDU

Os exemplos a seguir foram feitos usando a aplicação desenvolvida. Cada um dos campos é explicado em sua forma em hexadecimal, seguindo as informações disponibilizadas pela ISO 7816.

4.1.5.1 Exemplo 1:*SELECT DF*

O campo de CLA determina o tipo de instrução a ser interpretada e tem o seguinte significado:

- O *bit* b8 determina se o comando INS é proprietário ou não. Neste caso, b8 é 0, significando que é um comando intersetorial, reservado e definido pela ISO 7816-4.
- O *bit* b5 é determina se outros comandos devem ser esperados pelo emissor. Caso b5 seja 0, a INS é a última ou única instrução a ser enviada pelo emissor.
- Os *bits* b4 e b3 determinam se a mensagem deve utilizar um canal de segurança chamado *Secure Messaging*, Comunicação Segura (SH) que busca garantir confidencialidade de dados e autenticação de dados. A SH depende de um ou mais mecanismos de segurança aplicados que protegem todos ou parte dos dados enviados pela mensagem APDU
- Os *bits* b2 e b1 são usados para determinar um canal de comunicação lógico que não será abordado neste trabalho. A Figura 11 ilustra de forma simples a estrutura e os significados de cada *bit*, para um CLA padrão.

Figura 11 Tabela de bits de um comando *CLA*

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	x	-	-	-	-	**Command chaining control** (see 5.3.3)
0	0	0	0	-	-	-	-	— The command is the last or only command of a chain
0	0	0	1	-	-	-	-	— The command is not the last command of a chain
0	0	0	-	x	x	-	-	**Secure messaging indication**
0	0	0	-	0	0	-	-	— No SM or no indication
0	0	0	-	0	1	-	-	— Proprietary SM format
0	0	0	-	1	0	-	-	— SM according to Clause 10, command header not processed according to 10.3.3.2
0	0	0	-	1	1	-	-	— SM according to Clause 10, command header authenticated according to 10.3.3.2
0	0	0	-	-	-	x	x	**Logical channel number from zero to three** (see 5.4.2)

Fonte: (ISO 7816-3, 2006)

Realizando um teste de acesso à arquivo, a fim de obter dados básicos da *tag*, assim tem-se, respectivamente em relação ao comando da Figura 12:

1. 00 definindo que estamos usando um comando não proprietário;
2. $A4$ Instrução SELECT, comandos P1 e P2 determinam o que deve ser selecionado;
3. 04 *Select by DF name*, como os dois últimos bits estão vazios, selecionará um diretório especificado no *body;*
4. 00 define um padrão de controle, nesse caso, selecionará a primeira ocorrência;
5. $0E$ é o tamanho do campo de dados;
6. $325041592E5359532E4444463031$ é uma mensagem em hexa decimal que traduzida diz "$2PAY.SYS.DDF01$" usada para obter informações básicas do cartão, como o tipo de kernel usado e o tipo de AID (*Lifecycle Integrity Inc*, [*S. d.*]) e book B da EMVCo;
7. 00 diz ao cartão que não há limite para dados retornados.

Figura 12 Comando APDU para teste 1

Fonte: (Elaborado pelo autor, 2025)

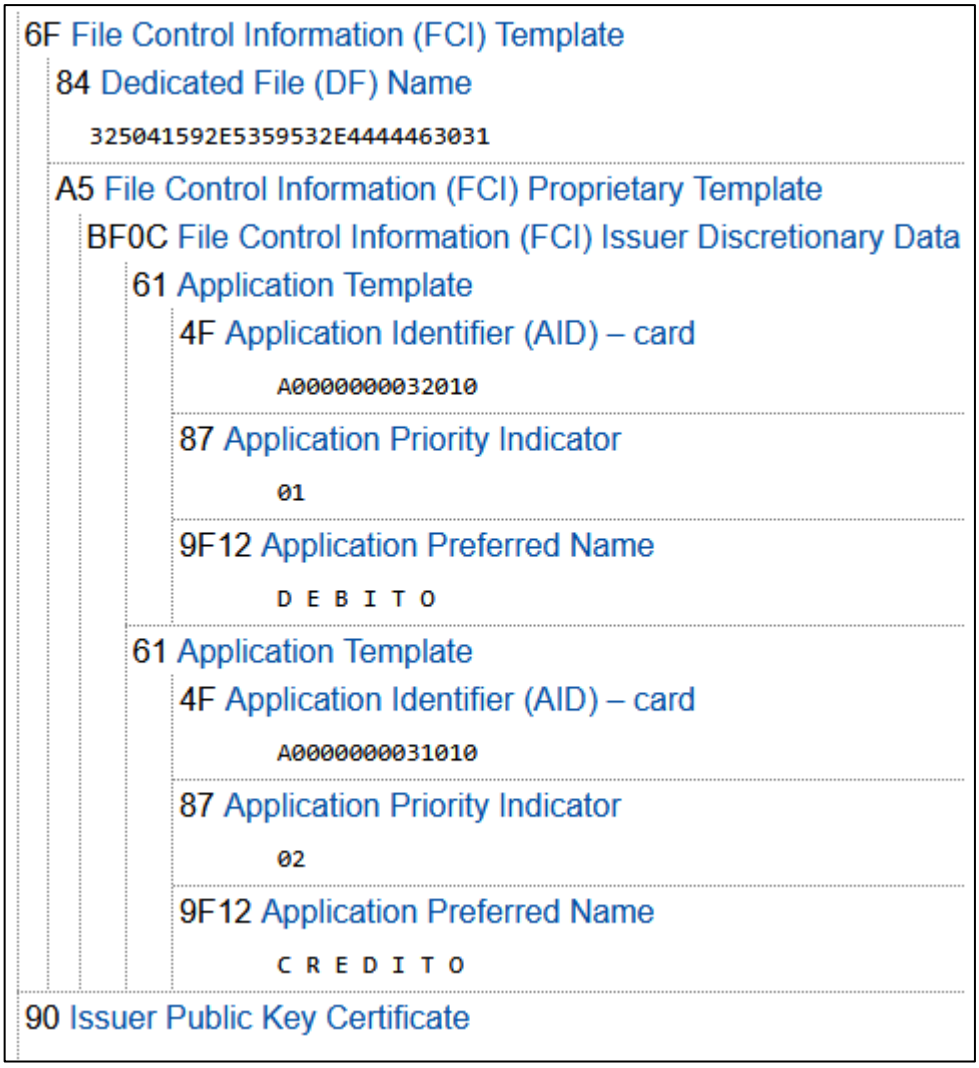
4.1.5.2 Exemplo 1:Retorno e interpretação de dados

Após enviar o comando, a *tag* NFC retorna a resposta a seguir:

```
6F3D8408A0000000030000000A5319F6E2A4830185782317202000B1141
015F3900521C000112920001129200011292 1B060000000113161B0700
009F6501FF9000
```

Usando a ferramenta *TLV Utilities* (EMVLAB, 2025)., essa resposta resulta nos dados mostrados na Figura 13:

Figura 13 Interpretação de resposta do exemplo 1



Fonte: (Elaborado pelo autor, 2025)

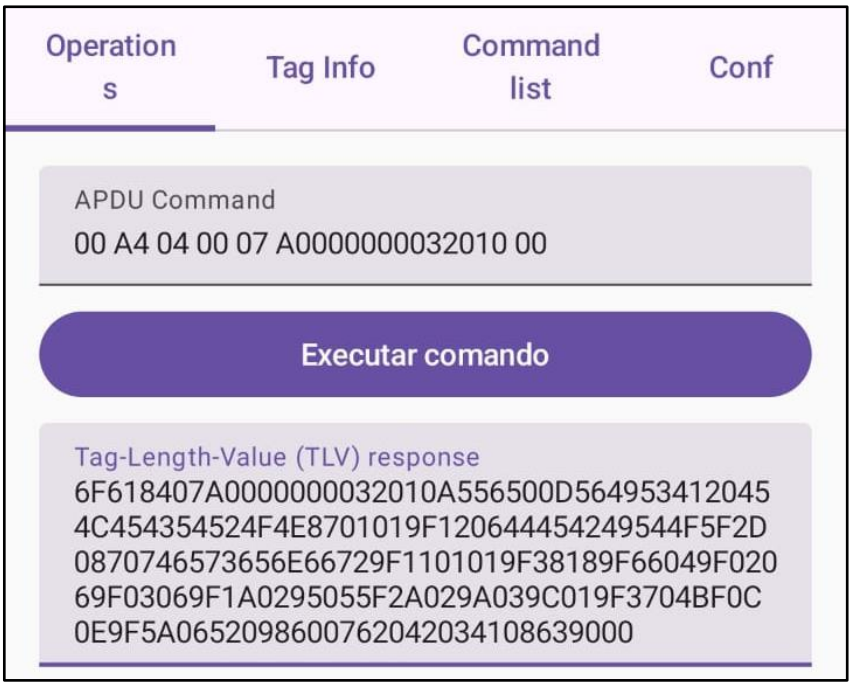
Dessa forma é possível obter os AIDs da *tag* NFC, permitindo outras operações mais complexas ao especificar as aplicações do dispositivo.

4.1.5.3 Exemplo 2: SELECT APPLICATION

Usando a AID do comando anterior, é possível obter mais informações de uma aplicação específica, o comando a seguir obtém metadados específicos da aplicação $A0000000032010$. O comando a seguir obtém mais metadados da aplicação específica de débito, usando uma estrutura similar do comando anterior, onde apenas o campo *body*, com o AID da aplicação.

Figura 14 contêm a tela da aplicação ao enviar o comando a *tag*.

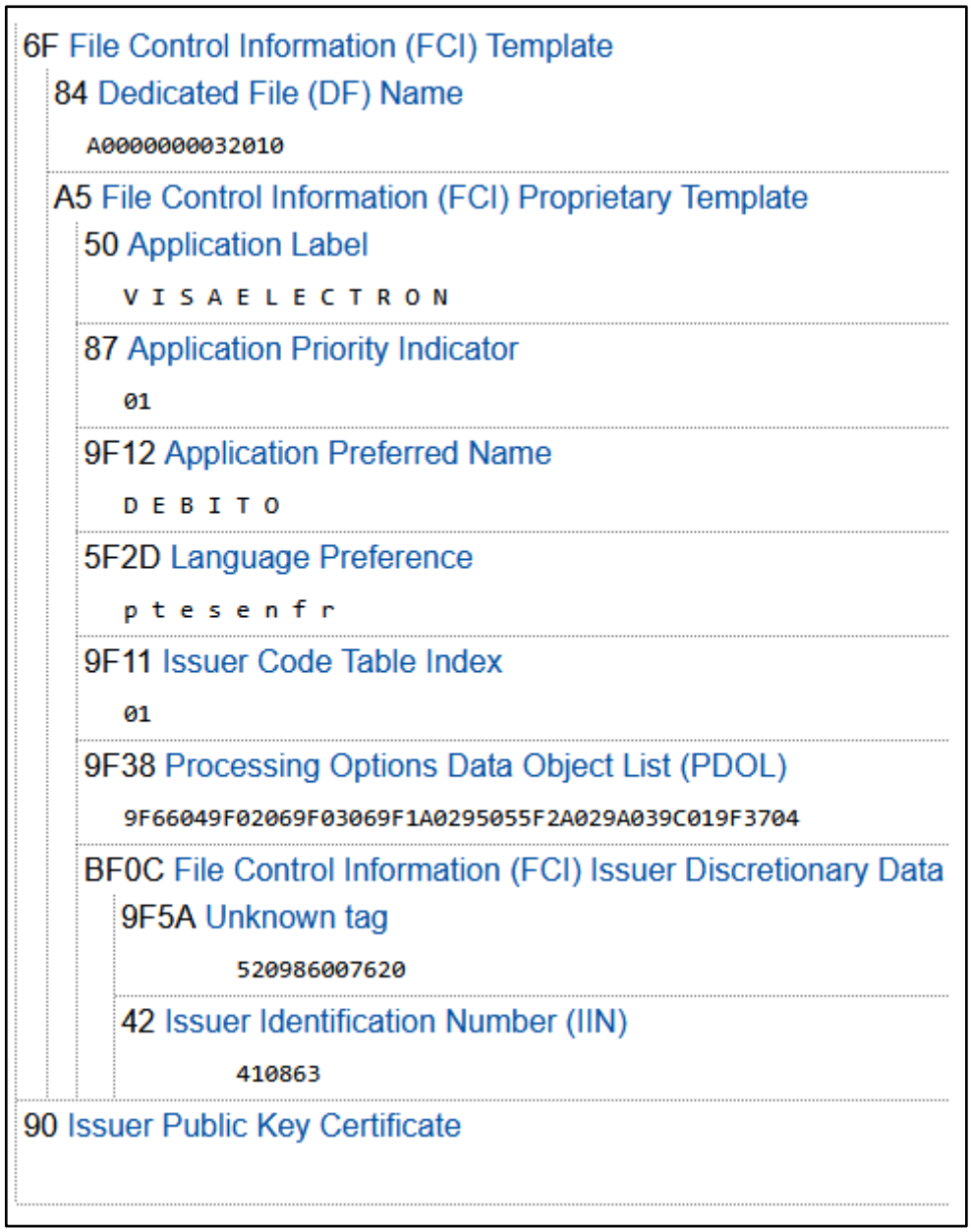
Figura 14: Comando APDU para teste 2



Fonte: (Elaborado pelo autor, 2025)

Figura 15 contém os dados de retorno interpretados. Algumas características dessa resposta é a Lista de Objetos de Dados de Opções de Processamento, *Processing Options Data Object List* (PDOL) que retorna ao terminal dados necessários para que qualquer transação ou coleta de informações sensíveis possa ser efetuada, com essas informações vaiando de acordo com cada emissora.

Figura 15: Interpretação de resposta do exemplo 2



4.1.6 Comentarios sobre testes da aplicação

Os testes realizados foram bem-sucedidos, confirmando a capacidade da aplicação em estabelecer a comunicação de baixo nível (usando especificamente IsoDep) com um smartcard e processar comandos conforme o padrão ISO 7816-4. O sucesso no envio e recebimento de bytes hexadecimais, juntamente com a decodificação das respostas, prova que a função *transceive()* está operando corretamente e que a aplicação é uma ferramenta funcional para comunicação básica.

O primeiro teste, SELECT DF, atingiu o objetivo de identificar as aplicações disponíveis (AIDs) na *tag*, fornecendo a base para operações subsequentes. O segundo teste, SELECT APPLICATION, demonstrou a capacidade da aplicação de direcionar comandos a uma aplicação específica (Débito ou Crédito, por exemplo) e obter informações cruciais para o fluxo de pagamento EMV.

Apesar do sucesso, os testes evidenciam a operação da aplicação em um modo de protocolo básico. A aplicação não provê o suporte para Secure Messaging (visto que todos os testes usam o APDU com CLA = 00) ou algoritmos de criptografia (como o BDH), mostrando mais um ponto de aprimoramento da aplicação, para que comandos mais complexos possam ser suportados, que dependem de um canal de segurança.

Outros testes de segurança se provaram sem sucesso devido a ausência dos mecanismos de segurança já discutidos. Por exemplo, o comando APDU $80\ A8\ 00\ 00\ 02\ 83\ 00\ 00$ que deve retornar valores como a chave púbica do cartão, só está disponível após o canal de segurança ser efetuado com sucesso (comentado no capítulo 3.2.1)

4.1.7 Processamento de comando enviado pelo usuário

A Figura 16 mostra a tela de envio e resposta para comandos APDU, internamente um objeto $NfcReaderViewModel$ é criado para gerenciar a conexão com a tag . Para enviar o comando, primeiro, transforma-se o código hexadecimal em binário e envia-se o comando usando a função $transceive()$ como mostra a Figura 7

Figura 16 Enviando comando ao cartão

```
result = withContext(Dispatchers.IO) {
try {
    if (!isoDep.isConnected) {
        isoDep.connect()
    }

    val responseBytes = isoDep.transceive(commandApdu)
    val responseString = "${bytesToHexString(responseBytes)}"
    Utils.d("Success response: $responseString")
```

```
    return@withContext responseString
} catch (e: IOException) {
    Utils.e("NFC IO Error: ${e.message}")
    return@withContext "Error: ${e.message}"
} finally {
```

Fonte: (Elaborado pelo autor, 2025)

4.2 Permissões de aplicativo e *AndroidManifest*

Por questões de segurança, a aplicação *mobile* deve requisitar permissão ao dispositivo para usar a tecnologia NFC no sistema isso é feito através do arquivo de $AndroidManifest.xml$. Também é necessário que a aplicação tenha, no mínimo, suporte ao SDK 10. Para requisitar acesso às funcionalidades de NFC, a linha a seguir deve ser adicionada no AndroidManifest.xml.

```
<uses-permission android:name="android.permission.NFC" />
```

Também é necessário "dizer" ao dispositivo quais tipos de tecnologias NFC a aplicação detecta. O sistema Android oferece três categorias, cada uma expandindo o suporte a diferentes classes de NFC, chamadas de *intents*:

Quadro 2: Tipos de *Intent*

Intent	Função
ACTION_NDEF_DISCOVERED	Suporte exclusivo a $NdefRecords$
ACTION_TECH_DISCOVERED	Suporte a tecnologias $NfcA$, $MifareClassic$ ou $IsoDep$
ACTION_NDEF_DISCOVERED	Usado caso as duas bibliotecas anteriores não tenham suporte para a *tag* detectada.

Fonte: (Android Developers, *s.d*)

A aplicação usada nesse trabalho usa a *ACTION_TECH_DISCOVERED* para suporte a classe *IsoDep*. O AndroidManifest.xml deve conter as seguintes linhas.

```
<Activity>
...
    <intent-filter>
        <action
    android:name="android.nfc.action.TECH_DISCOVERED"/>
    </intent-filter>
```

```
        <meta-data
        android:name="android.nfc.action.TECH_DISCOVERED"
            android:resource="@xml/nfc_tech_filter" />
        ...
</Activity>
```

Com isso, são declaradas as permissões necessárias para acessar os recursos NFC do dispositivo Android.

5 PROCEDIMENTOS METODOLÓGICOS

5.1 Ambiente de desenvolvimento

O ambiente de desenvolvimento consiste em um computador para desenvolvimento do código e um celular Android para testes

5.1.1 Computador

O projeto foi configurado com o Android *Software Development Kit* (SDK) versão 34 *(API Level* 34), assegurando compatibilidade com as versões mais recentes do sistema operacional. O processo de desenvolvimento ocorreu em um computador com sistema operacional Arch Linux, versão de kernel 6.17, processador *AMD Ryzen* 7 5700X 3.4 GHz, 16 GB de memória RAM.

5.1.2 IDE e Linguagem

O desenvolvimento do aplicativo foi realizado na *Integrated Development Environment*, Ambiente de Desenvolvimento Integrado (IDE) Android Studio, versão *Nahal* 4 2025.1.4, plataforma oficial disponibilizada pelo Google para o desenvolvimento de aplicações Android.

A linguagem de programação empregada foi o Kotlin 2.2.10, escolhida por ser a linguagem oficialmente preferida e primária para o desenvolvimento no ambiente Android (SOUSA, 2019). A linguagem que, por características técnicas tem integração nativa com o framework do sistema, possui vasta quantidade de material de referência e tutoriais, retrocompatibilidade com Java e é a linguagem padrão adotada pela

documentação oficial do Android, o que facilita o acesso a recursos e a resolução de problemas, agilizando o processo de desenvolvimento.

5.1.3 Dispositivo Android

Os testes de funcionamento e desempenho foram conduzidos em um dispositivo Xiaomi Poco 5x Pro, com Android 14, 8 GB de memória RAM e processador Octa-Core, a fim de observar o comportamento da aplicação em ambiente real. Complementarmente, foram utilizados os emuladores nativos do Android Studio para testar diferentes resoluções e versões do sistema.

5.1.4 Outras ferramentas de desenvolvimento

Durante o ciclo de desenvolvimento, foi empregado o sistema de controle de versão *Git*, com repositório hospedado no GitHub, para assegurar rastreabilidade e integridade do código-fonte. Adicionalmente, foram utilizados bibliotecas e frameworks do ecossistema *Android Jetpack*, como Jetpack Compose para a *interface* gráfica.

5.2 Etapas de desenvolvimento

5.2.1 Requisitos

Uma lista de requisitos funcionais e não funcionais da aplicação foi desenvolvida, conforme mostra a seguir:

5.2.1.1 Requisitos Funcionais:

- Ler *tags* NFC compatíveis (114443, ISO 15693, NDEF etc.);
- Enviar e receber comandos APDU via *Host-based Card Emulation*, Emulação de Cartão Baseada em *Host* (HCE);
- Estabelecer comunicação *peer-to-peer* entre dispositivos (modo Android Beam substituído por alternativas modernas);
- Identificar o tipo de *tag* detectada (*MIFARE*, *DESFire* etc.).

5.2.1.2 Requisitos Não Funcionais:

- Compatibilidade mínima com Android 10 ou superior;
- Interface simples e responsiva (Material *Design*);
- Processamento em tempo real e baixa latência na troca de comandos;
- Suporte a múltiplos idiomas (opcional).

5.2.1.3 Técnicas de desenvolvimento

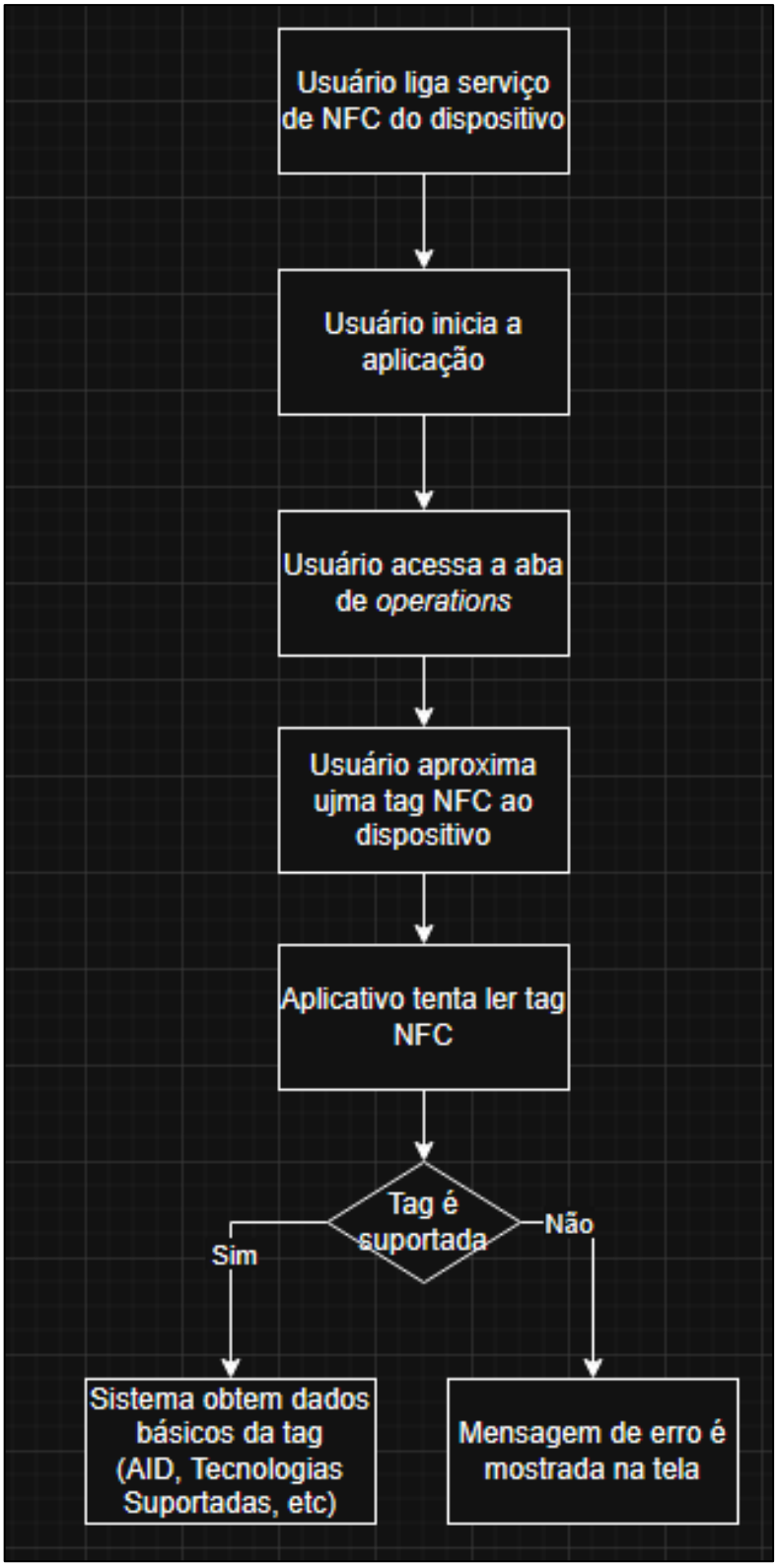
O padrão de arquitetura *Model–Model–View–Controller*, Modelo-Modelo-Visão-Controlador (MMVC) é uma variação do clássico MVC, aplicada ao desenvolvimento Android para melhorar a separação de responsabilidades e a testabilidade do código. Nessa abordagem, o primeiro *Model* representa as entidades de domínio e regras de negócio, enquanto o segundo *Model* (às vezes chamado de *ViewModel* ou *State Model*) atua como um intermediário entre a lógica e a *interface*, mantendo o estado da aplicação e evitando dependências diretas entre *View* e *Controller*. O *View* exibe dados e reage a eventos do usuário, e o *Controller* (geralmente uma *Activity* ou *Fragment*) coordena as interações, chamando métodos dos *Model*os e atualizando as visualizações.

. Segundo Steve Smith, (SMITH, 2024), padrões arquiteturais como o MVC e suas derivações permitem “[...] dimensionar o aplicativo em termos de complexidade, porque é mais fácil de codificar, depurar e testar algo (modelo, exibição ou controlador) que tem um único trabalho”. O MMVC segue esse princípio, adaptando-o ao ecossistema Android onde a persistência de estado e o ciclo de vida das telas devem ser constantemente monitorados, sendo uma alternativa intermediária entre o MVC tradicional e o MVVM adotado pelo *Android Jetpack*.

5.2.2 Fluxograma

A Figura 17 apresenta o fluxograma de uso básico da aplicação.

Figura 17: Fluxograma básico da aplicação



Fonte: (Elaborado pelo autor, 2025)

5.2.3 Telas de prototipagem

A Figura 18 representa uma tela de design dedicada a *logs* do sistema, feita na aplicação *Figma*. Essa tela foi descontinuada ao longo do projeto.

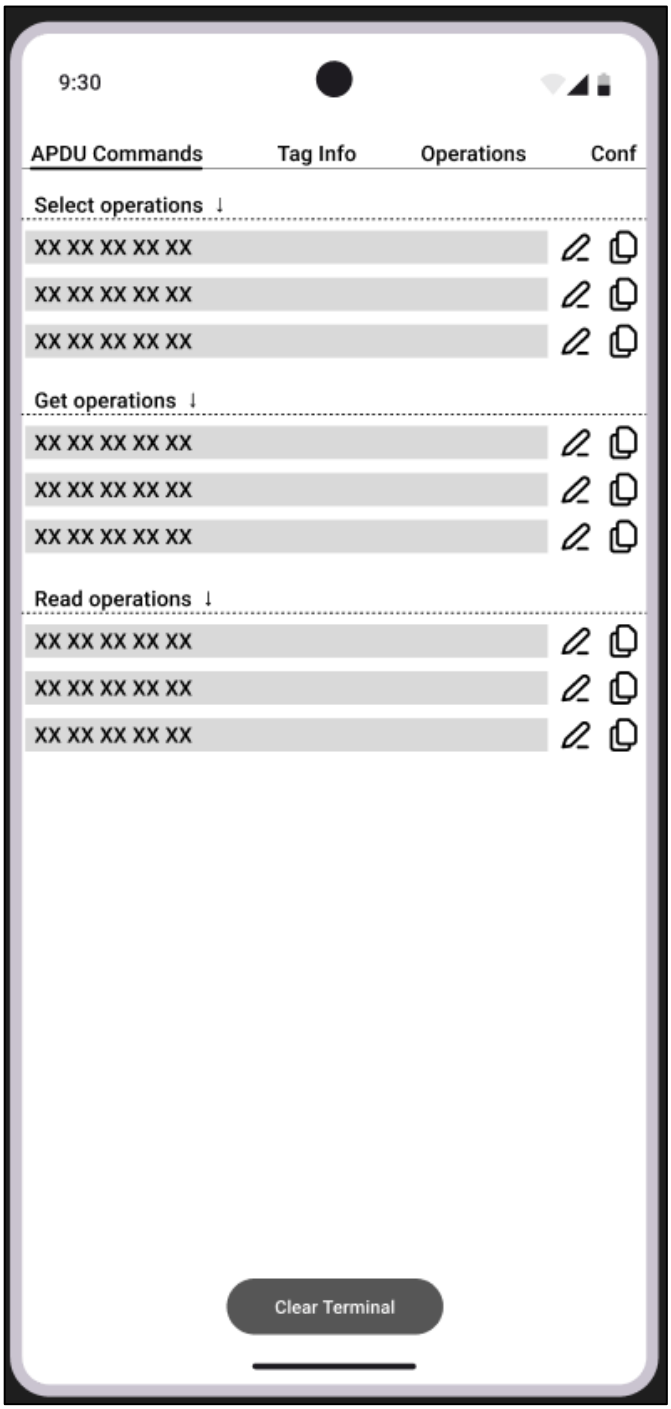
Figura 18: *Design* inicial da tela principal

Fonte: (Elaborado pelo autor, 2025)

Figura 19 representa uma tela de comandos básicos editáveis pelo usuário, permitindo que alguns comandos pré-definidos possam ser copiados rapidamente. Inicialmente, o usuário também poderia enviar os comandos em sucessão, algo como

encadear vários comandos com um único botão. Essa funcionalidade de edição de comentários foi descartada.

Figura 19 Tela de APDUS



Fonte: (Elaborado pelo autor, 2025)

5.2.4 Documento de classes

As classes relacionadas à *User Interface*, *Interface* de Usuário (UI) não foram adicionadas neste documento de classes. Quadros 3 e 4 definem as principais classes usadas no código da aplicação.

Quadro 3: Documento da classe *NfcReaderViewModel*

Classe		
Nome da classe	Descrição	
NfcReader*ViewModel*	Responsável por manter o estado da tela de leitura NFC ($NfcReaderScreen$), processar tags NFC, enviar comandos APDU via IsoDep e atualizar o estado da UI conforme regras de negócio.	
Atributos		
Nome	Tipo	Descrição
$\_uiState$	$MutableStateFlow < NfcReaderUiState >$	Estado interno mutável da UI, atualizado pela *ViewModel*.
$uiState$	$StateFlow < NfcReaderUiState >$	Estado exposto somente leitura para a UI.
$tagIsoDep$	$IsoDep?$	Referência à tecnologia IsoDep da *tag* atual.
$tagNfcA$	$NfcA?$	Referência à tecnologia NfcA da *tag* atual.
$readedTag$	$NfcReaderUiState?$	Último estado de *tag* lida, armazenado internamente.
Métodos		
Nome	Descrição	
onTagScanned(*tag*: *Tag*)	Processa uma *tag* NFC recém-lida, extrai informações e atualiza o estado da UI.	
ByteArray.toHexString(): String	Converte um *array* de *bytes* para uma string hexadecimal.	
sendMessageIso(hexCommand: String)	Envia um comando APDU via IsoDep, gerencia conexão e atualiza a resposta no estado da UI.	
checkIfValidHex(hex: String): Boolean	Verifica se uma string é válida no formato hexadecimal.	

Fonte: (Elaborado pelo autor, 2025)

Quadro 4: Documento da classe *MainActivity*

Classe		
Nome da classe	Descrição	
$MainActivity$	*Activity* principal do aplicativo. Gerencia ciclo de vida relacionado à captura de intents NFC, habilita e desabilita o *foreground dispatch* e conecta a UI (Compose) ao *ViewModel* responsável pela leitura NFC.	
Atributos		
Nome	Tipo	Descrição
$nfcAdapter$	$NfcAdapter?$	Adaptador NFC do dispositivo, usado para registrar e receber eventos NFC.
$nfcReaderViewModel$	$NfcReaderViewModel$	*ViewModel* utilizado pela *Activity* para processar tags NFC. Criado via by $ViewModels()$.
Métodos		
Nome	Descrição	
$onCreate(savedInstanceState: Bundle?)$	Inicializa a *Activity*, obtém o adaptador NFC e configura a tela principal com Jetpack Compose.	
$onResume()$	Ativa o *foreground dispatch*, garantindo prioridade ao app na leitura de tags NFC.	
$onPause()$	Desativa o *foreground dispatch* ao pausar a *Activity*.	
$onNewIntent(intent: Intent)$	Recebe intents NFC e envia a *tag* para o *ViewModel* processar, caso seja um tipo suportado.	

Fonte: (Elaborado pelo autor, 2025)

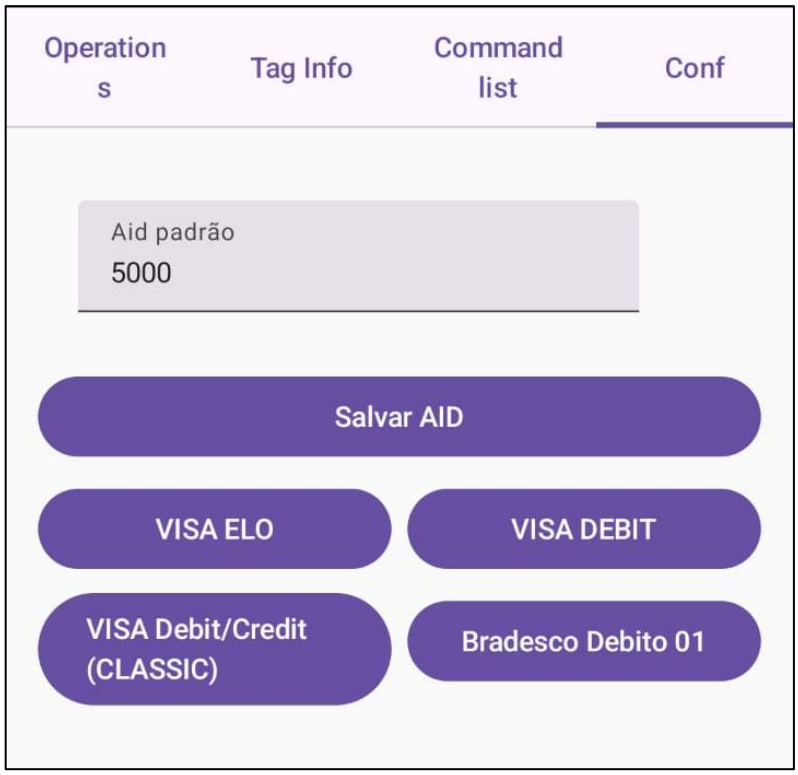
5.2.5 Telas da aplicação

Esse tópico trata das telas desenvolvidas para a aplicação, todas em sua versão final.

5.2.5.1 Tela de configurações

Figura 20 ilustra a tela de configurações, nela o usuário pode salvar uma AID que pode ser usada em outros comandos APDU. Inicialmente o App continha uma funcionalidade que permitia adicionar automaticamente essa AID em comandos APDU específicos, essa funcionalidade foi removida ao longo do desenvolvimento da aplicação.

Figura 20: Tela de configurações



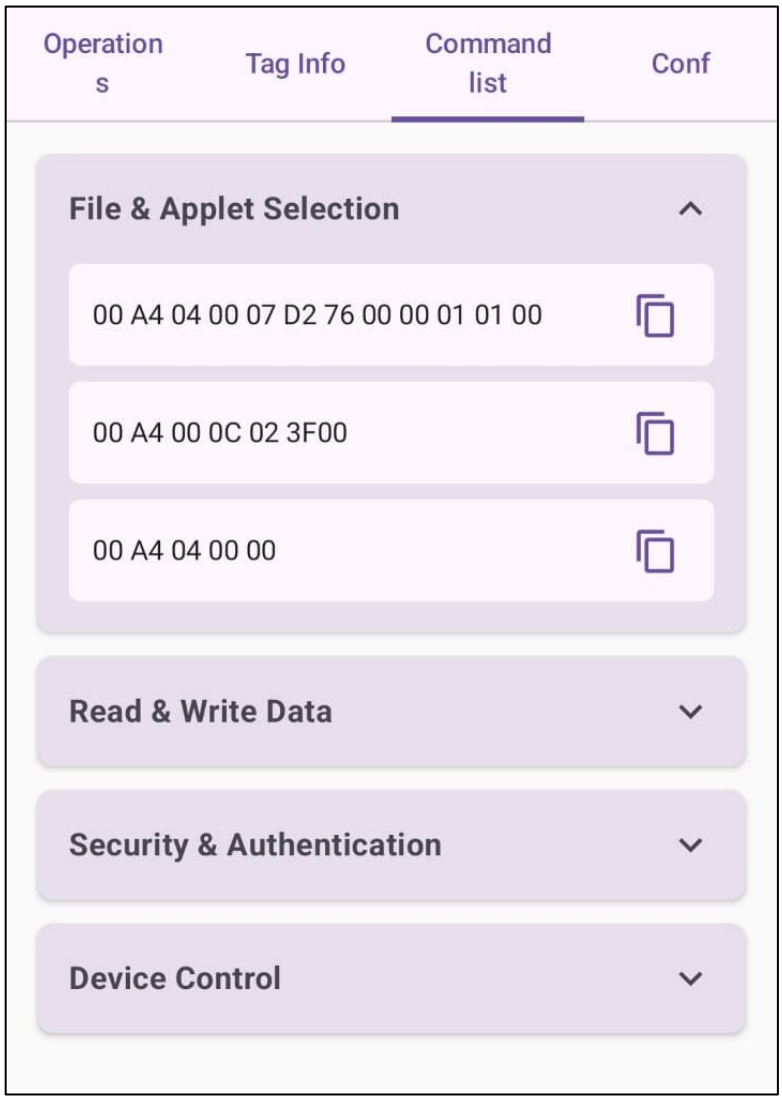
Fonte: (Elaborado pelo autor, 2025)

5.2.5.2 Tela de comandos básicos

Figura 21 ilustra a tela de comandos APDU, uma tela de comandos básicos, usados para testes. Esses comandos foram obtidos usando documentações oficiais

como a ISO 7816-3, *blogs* (Kaushik, Sourabh, 2023) e outros fóruns *online* que discutem sobre a tecnologia NFC. Alguns desses comandos não foram testados.

Figura 21: Tela de lista de comandos

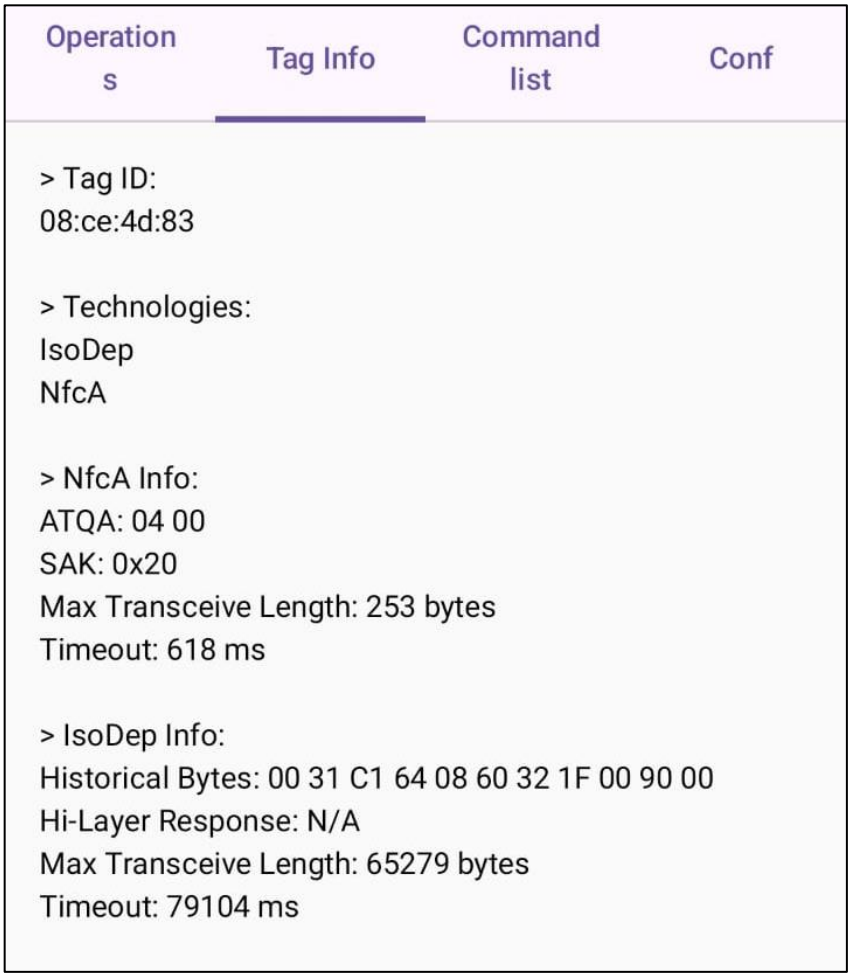


Fonte: (Elaborado pelo autor, 2025)

5.2.5.3 Tela de informações da *tag*

A Figura 22 ilustra interface de visualização dos metadados da *tag* NFC lida. Essa leitura é implementada usando as classes *IsoDep* e *NfcA*. A funcionalidade é detalhada no tópico 4.1.2 desse trabalho.

Figura 22: Tela de informações da *tag*

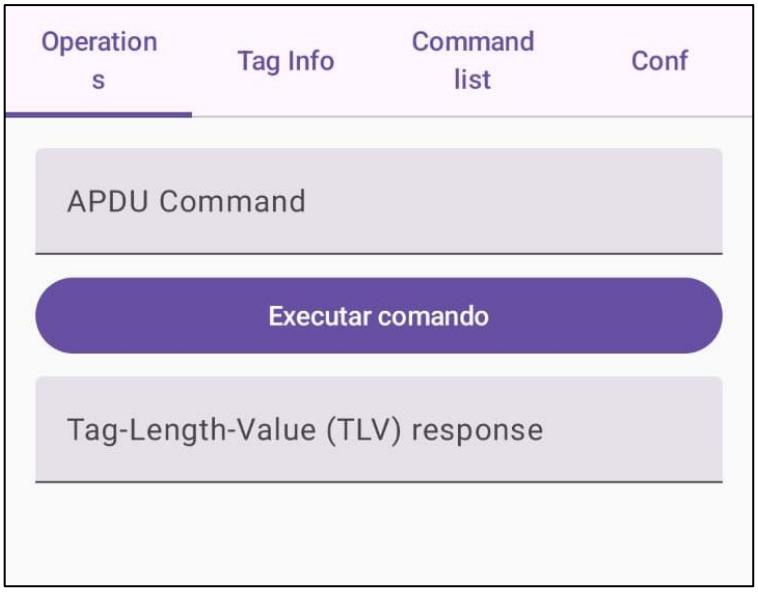


Fonte: (Elaborado pelo autor, 2025)

5.2.5.4 Tela para enviar operações

Figura 23 ilustra a interface de envio de comandos. Essa funcionalidade é detalhada no tópico 4.1.2 desse trabalho.

Figura 23: Tela de operações



Fonte: (Elaborado pelo autor, 2025)

5.3 Comentários sobre aplicação

O desenvolvimento da aplicação envolveu principalmente 3 principais, *Git*, para a organização do projeto, *Figma* para prototipagem e a ferramenta Android Studio.

Quanto a plataforma Android e desenvolvimento em geral, foi um grande desafio lidar com todas as peculiaridades do ambiente Android e da linguagem *Kotlin*, algumas delas sendo gerenciar o *Life Cycle* da aplicação corretamente, lidar com as permissões de usuário etc.

Quanto a aplicação em si, é evidente que muitas das funcionalidades panejadas para a versão final dela foram removidas. Seja por complexidade, habilidade durante as etapas de organização do código ou pela quantidade de erros de compatibilidade entre as funcionalidades.

Outras dificuldades incluem complexidade com a documentação EMV e ISO, tanto pela quantidade massiva de documentos disponíveis e, no caso da ISO, o fato de que muitas das documentações mais recentes não serem gratuitas ou de livre acesso.

Mesmo que os objetivos gerais e específicos foram alcançados, ainda sim a aplicação tem muito que melhorar, seja no âmbito de UI, facilidade de uso e funcionalidades importantes que ampliem o uso da tecnologia.

6 CONSIDERAÇÕES FINAIS

Com a constante adoção de sistemas de pagamento via NFC é necessário estudar como esses sistemas são implementados manualmente e como é o suporte da tecnologia de RF para plataformas Android. Este trabalho teve como objetivo principal analisar como ferramentas de leitura NFC podem ser implementadas em sistemas *Android* e tentar analisar como essas implementações são feitas.

Os objetivos propostos nesse trabalho foram alcançados, pois foi possível enviar e receber comandos entre o emissor e a *tag* NFC e desenvolver um código simples, mas suficientemente estável para enviar comandos para uma *tag* com suporte *IsoDep*.

Quanto aos procedimentos metodológicos, foram realizadas pesquisas com o objetivo de analisar como sistemas NFC são implementados, de forma básica, em dispositivos *Android* enviando comandos APDU para uma *tag* NFC (EMVco) e analisando as respostas recebidas. Para isso diversas documentações foram consultadas, ISOs (7816, 14443) documentos *NFC Forum* e diversos livros da série EMV Co.

Os principais desafios encontrados foram lidar com o *paywall* da série de especificações ISO, lidar com a linguagem de programação *Kotlin*, com a documentação muitas vezes confusa EMVCo e ISO, e, por fim, lidar com a falta de conteúdo avançado relacionado ao desenvolvimento de sistemas NFC. Sem funcionalidades presentes em um sistema POS (canal seguro, chaves de criptografia etc) padrão, obter qualquer tipo de informação relevante do cartão está além da capacidade deste trabalho, principalmente pela complexidade de implementação.

O artigo de Max Jakob (2015) mostra uma fundação para implementação de aplicações NFC usando a linguagem Java e outras classes não oficiais através do aplicativo *NFC Gate,* que implementa funcionalidades como operações de coleta de pacotes e *replay* de sessões de autenticação, e a matéria de Lakshmanan (2024) que explica como atacantes podem modificar uma aplicação de código livre apara realizar ataques maliciosos, não por si só, mas integrando ela com outros sistemas e outras vulnerabilidades.

O principal aprendizado desse trabalho foi em analisar como a tecnologia é implementada NFC, os mecanismos de segurança exigidos para que o sistema entre em produção e mostrar que um conhecimento aprofundado da tecnologia é necessário

para extrair qualquer tipo de informação dos dispositivos. Também foi de grande valor o contato com o ambiente *Android* e a linguagem *Kotlin*.

As contribuições deste trabalho para a sociedade são relevantes ao mostrar como esses sistemas de comunicação funcionam internamente, mesmo que de forma superficial

Com a aplicação desenvolvida, e os estudos realizados, foi possível enviar, receber e interpretar dados de uma *tag* NFC. Apartir desse conhecimento e os testes realizados, não foi possível encontrar meios para obtenção de informações pessoais do usuário usando apenas comandos e informações disponibilizadas nas documentações ISO, EVM e Android.

6.1 Limitações

Mesmo que este trabalho tenha conseguido alcançar os objetivos propostos ainda sim existem diversas limitações que devem ser reconhecidas:

- O aplicativo desenvolvido dá suporte apenas a operações básicas para dispositivos APDU, não permitindo múltiplos canais de comunicação;
- A aplicação não implementa algum tipo de criptografia durante o envio dos dados, funcionalidade que permita que o sistema de criptografia do dispositivo NFC via configuração de *bit* do campo CLA do comando, não implementa um decodificador nativo dos comandos APDU, tornado o processo de *debug* mais lento;
- O projeto usou de um cartão vencido do próprio pesquisador, seria interessante usar um cartão de crédito, funcional, dedicado a testes.

6.2 Sugestão de trabalhos futuros

- Implementar outras classes, especialmente com suporte a outros tipos de tecnologia NFC, sendo alguns os padrões *JIS*, *MiFare* e *NFC Forum*;
- Talvez utilizar algum dispositivo com root e tentar modificar, por meio de *software*, o funcionamento dos sensores NFC contidos no Android;
- Tentar analisar o código fonte de aplicativos oficiais de provedoras ou bancos digitais;

- Data a quantidade de informação espalhada na *Internet*, diversas vezes oculta de alguma forma, seria interessante um trabalho que foque e crie um centralizador de informações relacionadas a tecnologias NFC.

REFERÊNCIAS

AKTER, Sajeda *et al*. **Man-in-the-Middle Attack on Contactless Payment over NFC Communications: *Design*, Implementation, Experiments and Detection**. IEEE Transactions on Dependable and Secure Computing, New York, v. 18, n. 6, p. 3012-3023, 2021. DOI: 10.1109/TDSC.2020.3030213. Acesso em 26 mai. 2025.

Android. ***Near Field Communication (NFC)***. [*s. l*] *s.d*. Disponível em: https://developer.android.com/develop/connectivity/nfc/nfc. Acesso em: 10 set. 2025.

BANCO CENTRAL DO BRASIL. **Boxe 7 - Evolução de meios digitais para realização de transações de pagamento no Brasil. Relatório de Economia Bancária**. Banco Central do Brasil, 2022. p. 129-136. Disponível em: https://www.bcb.gov.br/content/publicacoes/boxe_relatorio_de_economia_bancaria/reb2022b7p.pdf. Acesso em: 26 mai. 2025.

BASUMALLICK, Chiradeep. **What Is NFC (Near Field Communication)? Definition, Working, and Examples**. [*s. l.*], 22 set. 2022. Disponível em: https://www.spiceworks.com/tech/networking/articles/what-is-near-field-communication. Acesso em: 13 mar. 2025.

BUSINESSWIRE. **Visa Tap to Phone Adoption Soars: 200% Year-over-Year Growth Worldwide**. [*s. l*] 3 mar. 2025. Disponível em: https://www.businesswire.com/news/home/20250303836483/en/Visa-Tap-to-Phone-Adoption-Soars-200-Year-over-Year-Growth-Worldwide. Acesso em: 21 maio 2025.

ELAD, Barry. **NFC Payment Statistics 2025: Understanding the Surge in Contactless Payments**. Coinlaw. [*s. l.*], 5 mar. 2025. Disponível em: https://coinlaw.io/nfc-payment-statistics. Acesso em: 27 maio 2025.

EMVCO. **EMV Contactless Specifications for Payment Systems**: Book A Architecture and General Requirements. [*S. l.*: *s. n.*], 2023. Disponível em: https://www.emvco.com/specifications/. Acesso em: 25 fev. 2025.

EMVCO. **EMV Contactless Specifications for Payment Systems**: Book E Security and Key Management. [*s. l..*: *s. n.*], 2025. Disponível em: https://www.emvco.com/specifications/. Acesso em: 26 fev. 2025

EMVCO. **EMV® Contactless Specifications for Payment Systems – Book E: Security and Key Management.** *[S.l.]*: EMVCo, 2023. Disponível em: https://www.emvco.com/wp-content/uploads/2024/01/EMVCo-Annual-Report-2023.pdf. Acesso em: 22 abr. 2025.

EMVCo. **WHAT are EMV® Specifications?.** [*S. l.*], [*S. d.*]. Disponível em: https://www.emvco.com/what-are-emv-specifications/. Acesso em: 23 abr. 2025.

EMVLAB. **TLVutils**. Disponível em: https://emvlab.org/tlvutils/. Acesso em 20 ago 2025

EverythingRF. **ResearchShows 85% of Consumers in Nine Countries Use NFC Technologies Everyday**. EverythingRF, [S. l.], p. 01-01, 28 jul. 2022. Disponível em: https://www.everythingrf.com/news/details/14852-research-shows-85-of-consumers-in-nine-countries-use-nfc-technologies-everyday. Acesso em: 21 mai. 2025.

GUTTMAN, Barbara; ROBACK, E. **An Introduction to Computer Security**: the NIST Handbook. [*S. l.*]: NIST Pubs, out. 1995. DOI https://doi.org/10.6028/NIST.SP.800-12r1. Disponível em: https://www.nist.gov/publications/introduction-computer-security-nist-handbook. Acesso em: 28 abr. 2025.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **ISO 7816-4:2020 – Identification cards – Integrated circuit cards – Part 4: Organization, security, and commands for interchange**. 4th ed. Geneva: ISO/IEC, 2020. Disponível em: https://www.iso.org/standard/77180.html. Acesso em: 25 abr. 2025

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **ISO/IEC 18092:2023 – Telecommunications and information exchange between systems — Near Field Communication Interface and Protocol 1 (NFCIP-1)**. 3rd ed. Geneva: ISO, 2023.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **ISO/IEC 21481:2021 – Information technology — Telecommunications and information exchange between systems — Near field communication interface and protocol 2 (NFCIP-2)**. 3rd ed. Geneva: ISO, 2021.

EMVCo. **EMVCo Members**. [*S. l.*], set. 2014. Disponível em: https://web.archive.org/web/20160215190249/http://www.emvco.com/about_emvco.aspx?id=156. Acesso em: 25 abr. 2025.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. *ISO/IEC 7816-3:2016* – **Identification cards – Integrated circuit cards – Part 3: Cards with contacts – Electrical interface and transmission protocols**. Geneva: ISO, 2006.

Kaushik, Sourabh. **Understanding APDU Commands: EMV Transaction Flow (Part - 2)**. Medium, 30 maio 2023. Disponível em: https://hpkaushik121.medium.com/understanding-apdu-commands-emv-transaction-flow-part-2-d4e8df07eec. Acesso em 17 out 2025

LAKSHMANAN, Ravie. **Ghost Tap: Hackers Exploiting NFCGate to Steal Funds via Mobile Payments**. The Hacker News, 20 nov. 2024. Disponível em: https://thehackernews.com/2024/11/ghost-tap-hackers-exploiting-nfcgate-to.html. Acessado em 10 nov 2025

Lifecycle Integrity Inc. **HOW TO READ NFC CARD DIRECTORIES**. [*S. d.*], Disponível em: https://lifecycleintegrity.com/how-to-read-nfc-dir/. Acessado em 27 out 2025

MAAß, Max Jakob; MÜLLER, Uwe; SCHONS, Tom; WEGEMER, Daniel; SCHULZ, Matthias. **NFCGate – An NFC Relay Application for Android**. In: ACM

Conference on Security and Privacy in Wireless and Mobile Networks (WiSec'15), 24–26 jun. 2015, New York. Proceedings […]. New York: ACM, 2015. doi: 10.1145/2766498.2774984. Disponível em: https://tubiblio.ulb.tu-darmstadt.de/146439/. Acessado em 15 out 2025.

PARDAL, Miguel; MARQUES, José. **Towards the Internet of Things: An Introduction to RFID Technology**. *In*: INTERNATIONAL WORKSHOP ON RFID TECHNOLOGY-CONCEPTS, APPLICATIONS, CHALLENGES, 4., 2010, Funchal. [*S. l.*], [*s. n*], 2010. p. 69-78. Disponível em: https://www.researchgate.net/figure/*Tag*-components-image-courtesy-of-Rafsec-OY-3_fig3_221311818. Acessado em 18 abr. 2025

PHILLIPS, Gavin. **How Does RFID Technology Work?**. [*S. l.*], 1 jun. 2017. Disponível em: https://www.makeuseof.com/*tag*/technology-explained-how-do-rfid-tags-work/. Acesso em: 18 abr. 2025.

RANKL, Wolfgang; EFFING, Wolfgang. **Smart Card Handbook**. 3. ed. [*S. l.*]: Wiley, 2004. ISBN 978-0470856680. Disponível em: https://books.google.com.br/books?id=ZurbCwAAQBAJ&lpg=PR5&dq=smart%20card&lr&hl=pt-BR&pg=PR3#v=onepage&q=smart%20card&f=false. Acesso em: 18 abr. 2025.

REVANKAR, Saisuman. **Apple Pay Statistics By Adoption, Usage, Demographic and Facts**. Coolest-Gadgets: Rohan Jambhale, 22 abr. 2025. Disponível em: https://www.coolest-gadgets.com/apple-pay-statistics. Acesso em: 27 maio 2025.

SECURETECHALLIANCE. **Nokia, Philips, and Sony establish the Near Field Communication (NFC) Forum**. https://www.securetechalliance.org, 18 mar. 2008. Disponível em: https://www.securetechalliance.org/nokia-philips-and-sony-establish-the-near-field-communication-nfc-forum/. Acesso em: 16 abr. 2025.

STALINGS, William. **Criptografia e Segurança de Redes: Princípios e Práticas**. 6. ed. [*S. l.*]: Pearson Universidades, 2014. 560 p. ISBN 978-8543005898.

STALLINGS, William; BROWN, Lawrie. **Segurança de Computadores**: princípios e práticas. 2. ed. [*S. l.*]: Elsevier, 1945. ISBN 9788535264494.

SOUSA, K. C. de. **Kotlin is now Google's preferred language for Android app development**. TechCrunch, [s. l.], 7 maio 2019. Disponível em: https://techcrunch.com/2019/05/07/kotlin-is-now-googles-preferred-language-for-android-app-development/. Acessado em 09 dez 2025:

THORAT, Neeta B.; LAULKAR, C. A. SURVEY ON SECURITY THREATS AND SOLUTIONS FOR NEAR FIELD COMMUNICATION. **International Journal of Research in Engineering and Technology**, [*s. l.*], v. 3, p. 291-295, 1 dez. 2014.

VIOLINO, Bob. **The History of RFID Technology**. [*s. l.*], 16 jan. 2005. Disponível em: https://www.rfidjournal.com/expert-*View*s/the-history-of-rfid-technology/76202/. Acesso em: 16 abr. 2025.

WARD, W. EMV card payments – An update. **Information Security Technical Report**, [*s. l.*], v. 11, ed. 2, p. 89-92, 2006. DOI https://doi.org/10.1016/j.istr.2006.03.001. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S1363412706000161?via%3Dihub. Acesso em: 18 abr. 2025.

WAZLAWICK, Raul Sidnei. **Metodologia de Pesquisa para Ciência da Computação**. 2 ed. Rio de Janeiro: Elsevier Editora Ltda, 2014. Acessado em: 17 abr. 2025

SMITH, Steve. **Visão geral do ASP.NET Core MVC**. *Microsoft Learn*. [*s. l*], 2024. Disponível em: https://learn.microsoft.com/pt-br/aspnet/core/mvc/overview?view=aspnetcore-10.0. Acessado em: 15 out 2025

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
GABINETE DO REITOR

Av. Universitária, 1069 ● Setor Universitário
Caixa Postal 86 ● CEP 74605-010
Goiânia ● Goiás ● Brasil
Fone: (62) 3946.1000
www.pucgoias.edu.br ● reitoria@pucgoias.edu.br

RESOLUÇÃO n° 038/2020 – CEPE

ANEXO I

APÊNDICE ao TCC

Termo de autorização de publicação de produção acadêmica

O(A) estudante Walker Freitas dos Santos
do Curso de Ciência da Computação ,matrícula 20222002800066 ,
telefone:_____e-mail walkerfreitassantos@gmail.com, na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado
_____
_____, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto (PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, ____ de _____ de _____.

Assinatura do(s) autor(es):_____

Documento assinado digitalmente
gov.br
WALKER FREITAS DOS SANTOS
Data: 07/10/2025 15:24:17-0300
Verifique em https://validar.iti.gov.br

_____

Nome completo do autor: _____

_____

Documento assinado digitalmente
gov.br
ANGELICA DA SILVA NUNES
Data: 18/12/2025 18:51:15-0300
Verifique em https://validar.iti.gov.br

Assinatura do professor-orientador: _____

Nome completo do professor-orientador: _____