Pontifícia Universidade Católica
do Rio de Janeiro

Rodrigo Motta Ferreira Corrêa

Reformulação de um sistema de monitoramento vestível baseado em IoT sob a óptica dos princípios da Segurança da Informação

Relatório de Projeto Final de Graduação II

Relatório de Projeto Final apresentado como requisito para obtenção do grau de Bacharel pelo programa de graduação em Engenharia de Computação da PUC-Rio

Orientador : Prof. Markus Endler
Co-orientador: Prof. Anderson Oliveira da Silva

Rio de Janeiro
Junho de 2023

Ao meu pai, José Carlos Gomes Corrêa,
e minha avó, Alda Motta Ferreira,
in memorian.

Agradecimentos

A Deus pela minha vida e pelas oportunidades de crescimento pessoal, espiritual e acadêmico.

A minha mãe pelo apoio, confiança e incentivo de sempre.

A todos os professores e funcionários do Departamento pelos ensinamentos e pela ajuda.

Ao autor do projeto o qual este se baseia, João Pedro Coutinho, por toda ajuda e disponibilidade durante o desenvolvimento deste trabalho.

Ao professor Anderson, por todo o auxílio na elaboração deste trabalho.

Aos meus colegas da PUC-Rio, BioBD e LampsCo.

A todos os amigos e familiares que de alguma maneira me estimularam ou me ajudaram a concluir este trabalho.

Resumo

Motta, Rodrigo; Endler, Markus; Oliveira da Silva, Anderson. **Reformulação de um sistema de monitoramento vestível baseado em IoT sob a óptica dos princípios da Segurança da Informação**. Rio de Janeiro, 2023. 40p. Relatório de Projeto Final II – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Este trabalho tem como objetivo repensar o desenho de um sistema de monitoramento de saúde baseado em IoT, levando em conta as principais vulnerabilidades de segurança da informação observadas e propondo implementações de controles adequados a cada vulnerabilidade observada. Isto é importante, uma vez que os dados de saúde monitorados são considerados sensíveis, indicando a necessidade urgente de um tratamento adequado na obtenção e transmissão destas informações ao longo do funcionamento do sistema.

Palavras-chave

Segurança da Informação; Internet das Coisas; Dados sensíveis; Vulnerabilidade; Ameaça.

Sumário

Lista de figuras

Lista de tabelas

Lista de Abreviaturas

ESB – *Enhanced ShockBurst*
SHW – *Smart Health Wearable*
SDK – *Software Development Kit*
SoC – *System on Chip*
PoC – *Proof of Concept*
MITM – *Man-in-the-Middle*
IoT – *Internet of Things*
UART – *Universal Asynchronous Receiver/Transmitter*
LGPD – Lei Geral de Proteção de Dados
ECDSA – *Elliptic Curve Digital Signature Algorithm*
AES – *Advanced Encryption Standard*
WPAN – *Wireless Personal Area Network*
WAN – *Wide Area Network*
LAN – *Local Area Network*
WEP – *Wired Equivalent Privacy*
WPA – *Wi-Fi Protected Access*
TLS – *Transport Layer Security*
AP – *Access Point*
PSK – *Pre-Shared Key*

1
Introdução

Não é incomum deparar-se com sistemas computacionais inseguros. Geralmente, a segurança cibernética dos sistemas é um requisito negligenciado nas etapas iniciais dos projetos de informática em função dos custos adicionais inerentes às implementações seguras. Em contrapartida, grande parte dos sistemas de computação lidam com dados sensíveis, sejam eles pessoais ou não, como senhas de acesso e dados de saúde. Nesse sentido, é imprescindível que sistemas como esses sejam planejados considerando os requisitos de segurança necessários para minimizar os riscos de exploração de vulnerabilidades. Os projetos de informática fundamentalmente seguros geralmente não exigem retrabalho, uma vez que já contemplam controles que diminuem os riscos envolvidos na utilização de um sistema computacional, sendo esta prática, conhecida como *security-by-design*, considerada ideal para a implementação robusta de um sistema computacional. Além disso, o estabelecimento de salvaguardas adequados contribuem para o alinhamento legal dos programas, facilitando a comercialização dos sistemas computacionais.

Como estudo de caso, este projeto pretende propor a implementação de controles de segurança da informação para o sistema *Smart Health Wearable* (SHW), com base em uma extensa análise sobre o mesmo. O sistema, assim como os descritos no parágrafo anterior, possui uma implementação limitada às funcionalidades de acompanhamento de saúde, não contando com controles de segurança no aferimento e transmissão dos dados de saúde envolvidos no projeto, que são por natureza sigilosos. Desta maneira, a proposta de reformulação apresentada neste relatório mostra-se como um estudo de caso de *redesign* sob a óptica da segurança da informação.

1.1
Objetivos

Além da responsabilidade de assegurar a privacidade de dados de usuários de sistemas computacionais, a Lei Geral de Proteção de Dados (LGPD), em vigor desde setembro de 2020 (BRASIL, 2018), lista uma série de obrigações e penalidades em caso de vazamento e/ou uso indevido de dados sensíveis, configurando uma importância na reformulação deste projeto com foco na segurança, já que os dados de saúde presentes ao longo do funcionamento do sistema são de caráter pessoal e consequentemente sigilosos. Minimizar os riscos de incidentes de segurança neste projeto, em especial os relacionados

com vazamento de dados pessoais, é essencial para que o mesmo seja viável em termos de conformidade legal e comercialização do produto.

Além disso, o projeto possui uma importância secundária de apresentar-se como um estudo de caso de reformulação de sistemas sob a óptica da segurança da informação, uma vez que não é incomum encontrar sistemas inseguros sendo utilizados em ambientes profissionais. Neste caso, a reformulação deste projeto demonstra, em segundo plano, uma oportunidade de aprendizado em como realizar uma intervenção significativa em um sistema já consolidado, visando a minimização dos riscos de segurança em projetos que envolvam Internet das Coisas (ou IoT — *Internet of Things*).

De forma mais específica, os objetivos do projeto são listados a seguir:

1. Identificar as vulnerabilidades e ameaças do SHW;
2. Mensurar qualitativamente o risco das ameaças identificadas;
3. Definir o nível de risco tolerável para o funcionamento considerado seguro do sistema;
4. Propor controles de segurança para minimização dos riscos, dentro dos limites de tolerância indicados;
5. Implementar os controles de segurança propostos ao SHW, visando o mínimo de retrabalho possível;
6. Verificar o funcionamento dos controles implementados;

1.2 Escopo da reformulação

A reformulação do projeto SHW considerará principalmente os conceitos de segurança da informação estudos ao longo do curso de graduação e durante a elaboração do projeto. Não serão realizadas melhorias relacionadas à requisitos funcionais do SHW, assim como também não serão considerados avanços sobre outros tipos de requisitos que não sejam relacionados com a minimização dos riscos de segurança do projeto original, como performance.

Em outras palavras, a prioridade deste projeto é propor a implementação de controles de segurança para o SHW e, se possível, implementá-los diretamente no código fonte original.

1.3 Metodologia

O SHW será repensado a partir da elaboração de uma análise de risco de segurança, que inicialmente definirá o espectro de probabilidades e impactos de possíveis explorações de vulnerabilidades do sistema, viabilizando a avaliação dos riscos que a ferramenta possui para cada ameaça.

Inicialmente, com base no contexto de utilização da ferramenta, serão definidas categorias de interesse da segurança da informação. Estas estarão relacionadas aos principais pontos de preocupação em relação aos riscos de segurança oferecidos pela utilização da ferramenta no estado atual, servindo como guia para o levantamento de vulnerabilidades que o sistema possivelmente apresente.

A partir das vulnerabilidades apontadas, serão listadas possíveis ameaças, i.e. explorações destas vulnerabilidades, atrelando a cada uma o risco que ela traz ao sistema. Este levantamento é importante porque auxiliará a definir quais riscos serão ou não tolerados no funcionamento do projeto em função de quais ameaças oferecem menor risco de segurança, seguindo um critério coerente com o contexto do sistema.

Finalmente, após o levantamento das ameaças e a definição da tolerância máxima dos possíveis riscos envolvidos na utilização do sistema, serão propostos salvaguardas adequados para cada ameaça intolerável. A implementação dos controles sugeridos tornará a utilização do projeto, dentro dos limites considerados aceitáveis, minimamente segura, idealmente viabilizando a conformidade legal e futura comercialização do sistema.

Vale ressaltar que o SHW já conta com pontos fortíssimos, como o baixo custo de implementação e facilidade de acompanhamento remoto do estado de saúde de pacientes. No entanto, o projeto foi concebido como prova de conceito (PoC - *Proof of Concept*), inviabilizando a elaboração de uma análise de risco enriquecida que contasse com mais possíveis participantes do projeto original, como patrocinadores, médicos e pacientes. A opinião destes atores sobre os riscos que o projeto possui e quais controles serão implementados para diminui-los é muito importante para eleger os controles de segurança adequados (JAGANNATHAN S., 2015), visto que a implementação dos salvaguardas não pode engessar a usabilidade do sistema e nem ter um custo inviável para a operação da ferramenta, mostrando a importância da participação de mais integrantes do projeto original na elaboração da análise de riscos.

2
Smart Health Wearable

O projeto *Smart Health Wearable* (SHW) foi um projeto concebido originalmente por João Pedro Coutinho em seu trabalho de graduação, apresentado-se como uma proposta de sistema de monitoramento de saúde não-invasivo e vestível, unindo eficiência operacional com baixo custo de implementação, uma vez que o projeto SHW baseia-se fortemente em dispositivos IoT (COUTINHO, 2021).

No entanto, o projeto foi originalmente pensado como um PoC, focando as implementações principalmente nas partes funcionais do sistema, deixando em segundo plano implementações seguras mais complexas, que contemplariam proteção na manipulação e transmissão dos dados pessoais sigilosos. Visando uma implementação comercial do SHW, um reprojeto focado em implementações seguras mostra-se essencial, uma vez que o controle adequado de dados pessoais de terceiros é uma obrigação legal, além de conferir mais confiabilidade ao sistema como um todo, o que certamente agrega para que o projeto tenha uma boa aceitação de mercado.

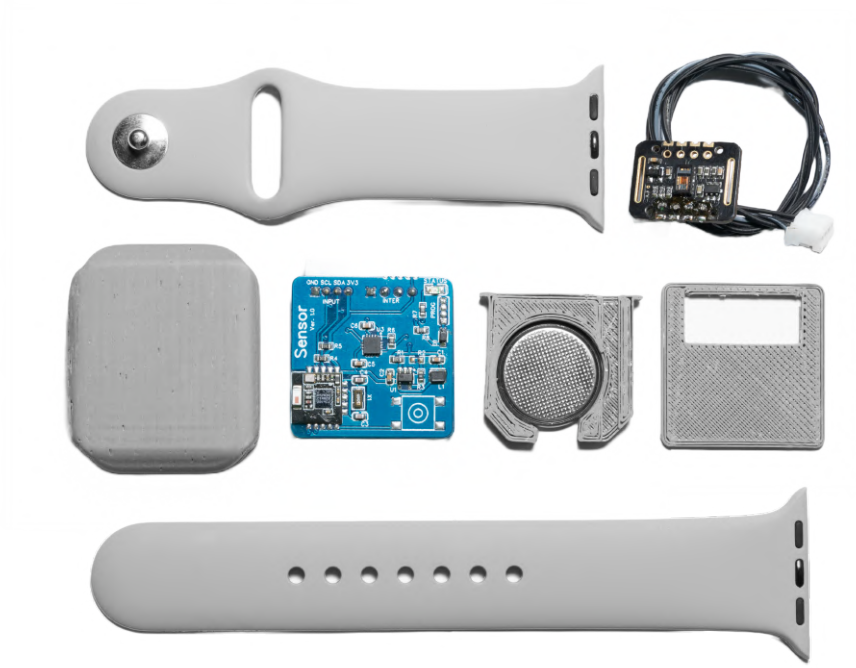

Figura 2.1: Amostra dos componentes do dispositivo vestível, responsável pela medição dos dados de saúde dos pacientes

2.1 Funcionamento geral

O SHW caracteriza-se por ser um sistema IoT, envolvendo a atuação de múltiplos dispositivos, cada um com sua responsabilidade. De forma geral, o sistema oferece aferimento e acompanhamento em tempo real da frequência cardíaca, temperatura corporal e taxa de oxigenação no sangue dos pacientes utilizadores. A medição é responsabilidade do *System-on-Chip* (SoC) nRF52832, fabricado pela *Nordic Semiconductor*, que fica atrelado a cada paciente em formato não-invasivo. Cada um dos dispositivos medidores se comunica via rádio com uma base centralizadora, utilizando o protocolo *Enhanced ShockBurst*, proprietário da fabricante do nRF52832.

A base é também implementada pelo SoC nRF52832, que recebe dados de saúde de até 8 medidores. Após o recebimento pelo nRF52832, o dispositivo encaminha os dados via protocolo UART (*Universal Asynchronous Receiver/-Transmitter*) para um dispositivo ESP8266, fabricado pela *Espressif Systems* e responsável pela comunicação com a *internet*.

Finalmente, o sistema prevê a comunicação da base centralizadora, através do ESP8266, com um servidor MQTT, utilizando Wi-Fi para conectar-se à *internet* e enviar os dados de saúde medidos através do protocolo citado. Idealmente, o sistema ainda prevê a utilização de um sistema processador destes dados, que obviamente se alimentará do servidor MQTT que recebe as mensagens, no entanto este não foi originalmente implementado. Um esquema ilustrativo do funcionamento geral pode ser visto na Figura 2.2.

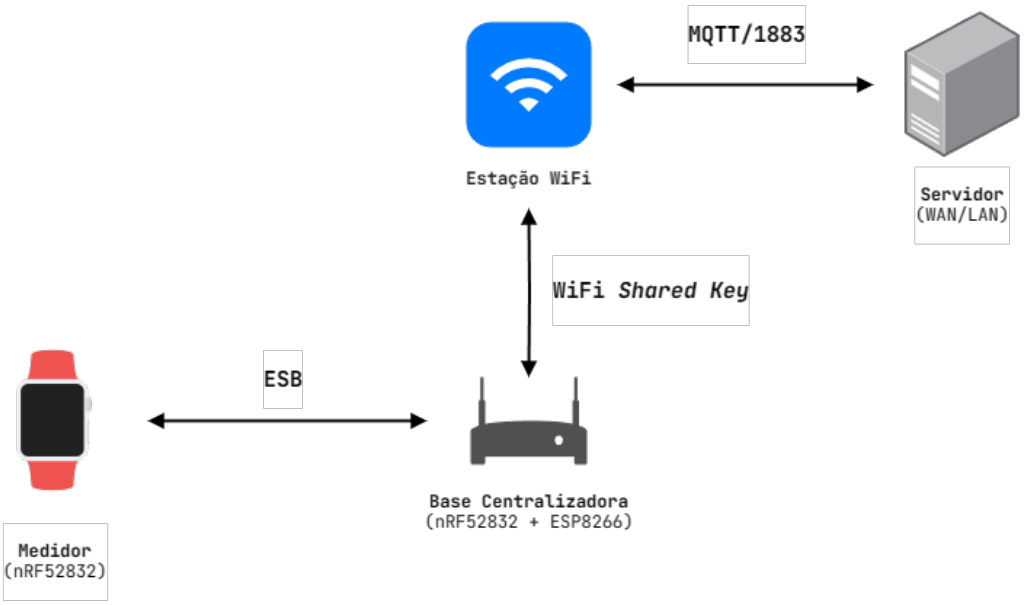

Figura 2.2: Grafo ilustrativo da transmissão de dados no *Smart Health Wearable*

2.2 Detalhes do funcionamento

O sistema SHW, na parte dos medidores e base centralizadora, foi originalmente programado utilizando a linguagem C++ com apoio da IDE *Segger Embedded Studio*, também da *Nordic Semiconductor*, e do *Software Development Kit* (SDK) oferecido pela fabricante para desenvolvimento nos dispostivos da série nRF52 (Nordic Semiconductor, 2020).

Já na parte do microcontrolador ESP8266 toda a lógica foi concebida utilizando a linguagem Lua, com apoio da IDE *ESPlorer*, proprietário da *Espressif Systems*, fabricante deste dispositivo.

As subseções seguintes detalham o funcionamento das tecnologias utilizadas no projeto original, destacando principalmente as questões de segurança mais evidentes em cada parte citada na descrição geral do funcionamento do sistema.

2.2.1 Comunicação de rádio entre dispositivos nRF52832 utilizando ESB

Cada dispositivo vestível, responsável pelo aferimento dos dados de saúde dos pacientes, comunica-se via rádio com uma base centralizadora utilizando o protocolo *Enhanced ShockBurst* (ESB). O protocolo é proprietário da Nordic, fabricante dos dispositivos envolvidos nesta transmissão de dados, e não possui uma documentação extensa, sendo um dificultador para avaliação de segurança do protocolo. Toda documentação disponível fornecida pela fabricante encontra-se no guia de usuário disponibilizado na documentação do kit de desenvolvimento para os SoC's da série nRF52.

Cada dispositivo receptor PRX, localizado na base centralizadora, consegue estabelecer comunicação com até 8 dispositivos transmissores, denominados PTXi e organizados em topologia de estrela, conforme demonstrado pela ilustração da Figura 2.3.

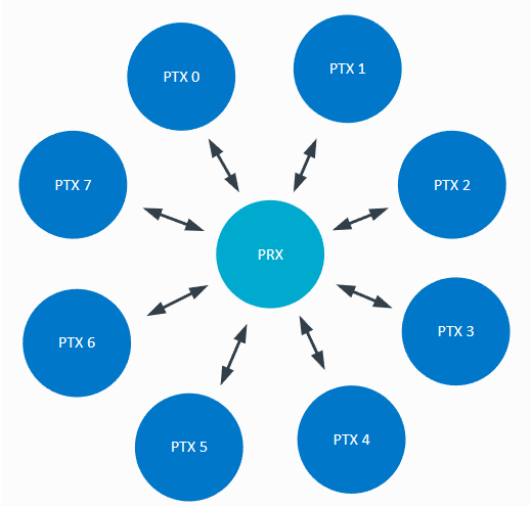

Figura 2.3: Demonstração fornecida pela Nordic da topologia de rede na comunicação entre os dispositivos da série nRF52 utilizando o ESB

De acordo com o guia de usuário fornecido pela fabricante, a transmissão inicia-se quando um dispositivo receptor aloca em uma fila de transmissão o conteúdo do pacote a ser enviado. Automaticamente, o transmissor envia o pacote ao dispositivo receptor que, caso receba-o com sucesso, devolve ao remetente um pacote do tipo *acknowledgement* (ACK), com o objetivo o recebimento do pacote de dados. A utilização do pacote ACK fica a cargo do programador, sendo ativada por padrão (Nordic Semiconductor, 2016b).

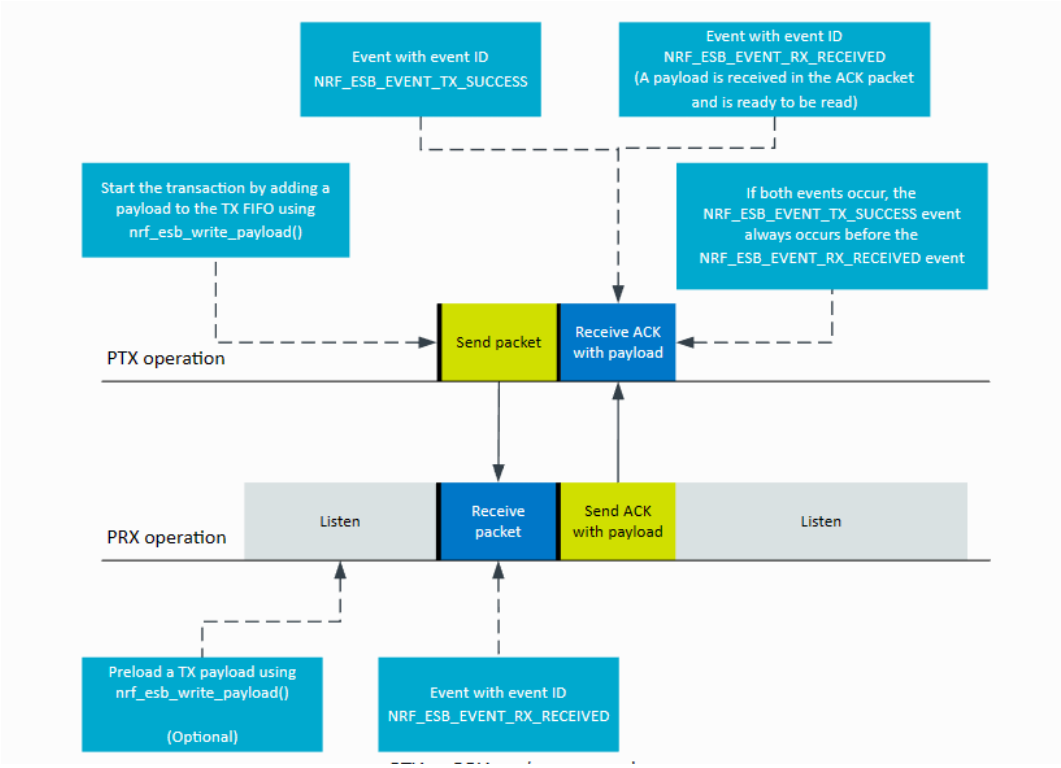

Figura 2.4: Linha do tempo de eventos no processo de transmissão de um pacote utilizando o ESB

O protocolo possui fácil implementação e funcionamento simples. No entanto, não possui nenhuma referência com relação à implementações seguras. Isto significa que não há nenhuma proteção de rádio na comunicação utilizando o protocolo, o que viabiliza ataques de *sniffing* ou *Man in the Middle* (MITM), por exemplo, onde o atacante pode analisar todo o tráfego de rádio corrente ou ainda forjar pacotes transmitidos.

Tal fato expõe uma violação grave do sigilo dos dados de saúde corrente na comunicação utilizando o protocolo ESB, mostrando-se ineficiente para uma implementação segura.

2.2.2 Aferimento e persistência de dados no nRF52832

Os dispositivos medidores vestíveis realizam persistência temporária em memória dos dados de saúde que falharam em retransmissões do ESB (COUTINHO, 2021). Em caso de falha, uma função gerenciadora de eventos do dispositivo transmissor guarda em um *buffer* o conteúdo do pacote não transmitido.

Este é um mecanismo importante, já que mantém os dados de saúde medidos mesmo que a transmissão não seja possível, impedindo a perda de dados de monitoramento dos pacientes. É razoável afirmar que a ação é

completamente natural considerando a natureza física da transmissão, o rádio, que normalmente é ruidosa, portanto uma rotina de salvamento temporário dos pacotes ajudaria a suavizar a perda de pacotes durante a transmissão.

Além disso, falhas de energia do transmissor ou falta de alcance são mitigadas a partir da utilização deste *buffer*, evitando que dados sejam perdidos e o paciente possa ser monitorado de forma remota, ainda que apresente um atraso na transmissão dos dados.

Entretanto, os dados persistidos, ainda que temporariamente, na memória do SoC, não possuem nenhum controle de integridade, autenticidade e sigilo, configurando uma situação de vulnerabilidade envolvendo os dados pessoais dos pacientes. Como solução, a melhor saída é implementar estes controles, sem abrir mão da funcionalidade de retenção temporária das medições, o que comprometeria o funcionamento original do projeto.

2.2.3 Utilização da internet no ESP8266 utilizando MQTT

Após o recebimento dos pacotes pelo dispositivo receptor nRF52832, os dados de saúde recebidos são transferidos, internamente na base, para o SoC ESP8266, responsável pela comunicação com a internet.

O ESP8266 comunica-se via *WiFi* através de padrão desconhecido, assumindo-se qualquer um, para fins de análise. Neste caso, a possibilidade de utilização do padrão WiFi *Wireless Equivalent Privacy* (WEP), considerado inseguro (JAGANNATHAN S., 2015) caracteriza mais uma situação de vulnerabilidade com necessidade urgente de solução. A utilização deste padrão mostra-se insegura pela forma de conexão e esquema de autenticação, que utiliza chaves compartilhadas. Portanto, em caso de vazamento da chave, um agente ofensivo poderia se passar como o *Access Point* (AP) do sistema e atuar como MITM no funcionamento do sistema, configurando uma falha no controle de sigilo destes dados também.

Além disso, a comunicação com a internet é estabelecida utilizando o protocolo *Message Queuing Telemetry Transport* (MQTT), que possibilita implementação segura, mas não possui controles de segurança por padrão (DINCULEANă D., 2019). Isto significa que toda a comunicação de internet realizada pelo sistema trafega na rede em texto plano, representando uma falha grave no controle de sigilo dos dados pessoais de saúde manejados ao longo do funcionamento do sistema, exigindo intervenção imediata.

3
Análise de risco

A Análise de risco de segurança é um documento importante no fundamento de qualquer sistema. Uma análise desse tipo mapeia categoricamente os pontos fracos do sistema analisado, atribuindo níveis de risco arbitrários coerentemente definidos, viabilizando o levantamento de sugestões de segurança adequadas para um funcionamento minimamente seguro que não inviabilize a operação e o financiamento do projeto (SILVA, 2018).

É importante ressaltar que nunca será possível garantir com absoluta certeza a segurança do funcionamento de nenhum sistema, sendo o principal objetivo definir quais riscos são toleráveis e quais riscos deverão ser controlados de forma mais rígida. Com isso, o esquema de segurança da informação de qualquer sistema computacional visa evitar ao máximo qualquer incidente de segurança, a partir de salvaguardas adequados.

Nesse sentido, a Análise de risco de segurança deste projeto utiliza as informações do funcionamento do sistema expostas no capítulo anterior para fornecer uma definição concreta de quais riscos o projeto está correndo, quais as principais vulnerabilidades encontradas, quais são as ameaças relacionadas a estas vulnerabilidade e, finalmente, quais riscos não serão tolerados no funcionamento do projeto, i.e., quais controles de segurança deverão ser implementados para um funcionamento minimamente seguro do sistema.

3.1
Foco da Análise

Esta Análise de risco terá enfoque nas seguintes categorias de segurança da informação:

1. Vazamento de dados pessoais
2. Ataques de código não-autorizado e/ou malicioso
3. Utilização não autorizada de recursos tecnológicos
4. Indisponibilidade de serviço intencional

3.2 Definição de Risco

A definição das faixas de probabilidade da ocorrência de ações maliciosas e impacto consequente das explorações é essencial para a Análise de risco. O conceito de risco, ainda que arbitrário, decorre do cruzamento entre estes dois conceitos, podendo ainda ser definido de forma matricial, relacionando explicitamente os impacto e probabilidades.

Com o objetivo de facilitar o entendimento, a definição de risco é realizada de forma não-matricial e está explicitamente descrita na Tabela 3.3.

Vale ressaltar que as definições apresentadas nesta seção são referenciadas em seções futuras, principalmente para definir os riscos toleráveis para o projeto.

Faixas de probabilidade:

Probabilidade	**Descrição**
Improvável	É esperado que a ameaça não seja explorada por nenhum atacante na maioria dos casos
Possível	Existe uma possibilidade razoável de que a ameaça seja explorada por um atacante
Provável	Existe uma chance alta de um atacante explorar a ameaça
Muito provável	A ameaça certamente será explorada por atacantes

Tabela 3.1: Definição dos níveis de probabilidade de exploração de uma vulnerabilidade

Faixas de impacto:

Impacto	**Descrição**
Baixo	Se explorada, a ameaça não compromete de nenhuma maneira o funcionamento normal do sistema, nem oferece risco de vazamento de dados
Médio	A ameaça não compromete os dados sensíveis e/ou o monitoramento, mas realiza ações não autorizadas que podem atrapalhar o sistema, aumentando levemente a chance de ocorrerem vazamentos ou inoperância no monitoramento
Alto	A ameaça pode comprometer o funcionamento normal do sistema, impedindo o monitoramento a partir de exploração de recursos não autorizada, com uma chance razoável de vazamento de dados sensíveis
Muito alto	A ameaça compromete o funcionamento integral do sistema, com chances reais de exposição indevida dos dados sensíveis armazenados

Tabela 3.2: Definição dos níveis de impacto consequente da exploração de uma vulnerabilidade

Faixas de risco:

Risco	Descrição
Baixo	Risco tolerável, sem necessidade de acompanhamento especial da ameaça
Médio	Risco tolerável, porém com chances razoáveis de causar um impacto significativo. Neste caso, a ameaça deve ser monitorada, sem urgência para a implementação de controles de segurança
Alto	Risco totalmente intolerável, no qual o impacto da ameaça e sua probabilidade não são ignoráveis. Ameaças nesta faixa exigem a implementação de salvaguardas imediatamente

Tabela 3.3: Definição dos níveis de impacto consequente da exploração de uma vulnerabilidade

3.3 Levantamento de vulnerabilidades e ameaças

O levantamento de possíveis vulnerabilidades e ameaças foi realizado de forma categórica, estabelecendo relação direta com a implementação atual do sistema.

Cada subseção representa uma das categorias definidas no início deste capítulo, das quais cada uma apresenta, em suas subseções, pontos de vulnerabilidade, que são em seguida relacionados com possíveis ameaças ao sistema e suas classificações de risco.

3.3.1 Vazamento de dados pessoais

Esta categoria é dedicada a identificar vulnerabilidades que permitam o acesso não autorizado aos dados de saúde coletados, que são de caráter pessoal e sensível. As vulnerabilidades observadas no sistema que permitem o vazamento de dados pessoais são listadas a seguir:

3.3.1.1 Persistência de dados

A persistência de dados que ocorre no *buffer* do dispostivo medidor nRF52832 é insegura, não contando com controles de integridade e sigilo, que são essenciais para a manutenção de dados pessoais sensíveis, como os

envolvidos no funcionamento do sistema.

Possíveis ameaças:

- ➤ Modificação não autorizada dos dados armazenados;
- ➤ Apropriação indevida dos dados armazenados via descarregamento de memória do dispositivo;

Classificação de risco das ameaças: considerando a natureza dos dados vazados, o impacto será muito alto caso alguma ameaça seja explorada. Dada a facilidade dos ataques para um agente externo, e até mesmo interno, a probabilidade de explorações que ocorrem também pode ser considerada alta, configurando um **risco ALTO.**

3.3.1.2 Comunicações sem fio

As comunicações de rádio do sistema não possuem qualquer controle de integridade, autenticidade e sigilo, comprometendo os dados que trafegam neste meio. Apesar do padrão WiFi utilizado ser desconhecido, a própria utilização do MQTT desprotegido também oferece risco de vazamento na comunicação entre base e servidor.

Possíveis ameaças:

- ➤ Interceptação dos dados via rádio e/ou internet/intranet;
- ➤ Modificação dos dados por agente externo malicioso intermediário;
- ➤ Falsificação da autoria dos dados medidos, permitindo o tráfego de dados falsos/inválidos no funcionamento comum do sistema;

Classificação de risco das ameaças: assim como as ameaças de persistência de dados em texto plano, por configurar uma possibilidade de vazamento de dados pessoais, **o risco é considerado ALTO**

3.3.2 Ataques de código malicioso

Esta categoria tem como objetivo identificar possibilidades de substituição não autorizada parcial ou total do(s) código(s) utilizado(s) na arquitetura do sistema. A injeção de código malicioso ou exploração de vulnerabilidades dentro do código pode comprometer dados e funcionamento, além de apresentar uma infinidade de possíveis ações maliciosas a partir dos dispositivos da arquitetura.

A seguir, são apresentados os fatores da arquitetura do projeto que podem levar à exploração de ameaças nesse sentido:

3.3.2.1 Implantação de código não autorizado

Os dispositivos não realizam nenhuma checagem de integridade do código, podendo executar potencialmente qualquer *script* que receberem, por quaisquer meios.

Possíveis ameaças:

- Realização de funcionalidades indevidas, como envio de dados medidos para agente externo malicioso ou utilização de recurso computacional para fins diversos não autorizados;

Classificação de risco das ameaças: pelo alto potencial de ataque que pode ser alcançado através destas ameaças, o impacto é considerado muito alto. Mesmo que a probabilidade desse ataque seja considerada média, em função do impacto, **o risco é considerado ALTO**, uma vez que, após uma invasão, potencialmente qualquer ação poderá ser realizada no sistema, inclusive roubar os dados pessoais, parte mais sensível e valiosa no funcionamento da ferramenta.

3.3.3 Utilização não autorizada de recursos

Esta categoria visa identificar utilizações não autorizadas do sistema, tanto física quanto digital. Encaixam-se aqui o extravio do aparelho e exploração de recursos computacionais indevida, por exemplo.

3.3.3.1 Utilizadores não identificados

A aplicação não consegue identificar que pessoa física está utilizando qual dispositivo medidor, impedindo a identificação de uma situação de extravio ou troca não autorizada do paciente que está sendo monitorado em um dado instante, podendo comprometer o funcionamento esperado do sistema

Possíveis ameaças:

- ➤ Identificação não apropriada dos dados de saúde ao paciente correto;
- ➤ Utilização do sistema por pessoas não autorizadas, interna ou externamente;

Classificação de risco das ameaças: esta vulnerabilidade possui probabilidade média de ocorrer em função dos ambientes que o sistema será utilizado que, ainda que sejam indefinidos pela documentação original, certamente terão um controle físico razoável (hospitais e domicílios, por exemplo), diminuindo as chances de extravio do aparelho medidor. O impacto também é considerado médio, uma vez que os dados sensíveis não correm risco direto de serem comprometidos neste caso. **Caracteriza-se portanto risco MÉDIO.**

3.3.4 Indisponibilidade de serviço

Finalmente, esta categoria é dedicada a identificar vulnerabilidades que, se exploradas, tornem o serviço indisponível, desabilitando o funcionamento do sistema, parcial ou integralmente.

3.3.4.1 Poluição maliciosa dos canais de comunicação

Nenhum controle de rede, nem de rádio, e nem de WiFi, está previsto na especificação do projeto, indicando uma possibilidade razoável de atividade maliciosa de poluição dos sinais rádio, bem como a inundação do tráfego WiFi, podendo aumentar a latência no recebimento dos dados entre as comunicações sem fio que podem até mesmo levar a inoperância total do sistema em termos de tráfego de dados ao longo de sua arquitetura.

Possíveis ameaças:

- Negação de serviço intencional;

Classificação de risco das ameaças: A probabilidade da exploração da ameaça pode ser considerada baixa, devido a operação provavelmente ocorrer em ambiente controlado. No entanto, caso ocorra, o impacto será alto, uma vez que o funcionamento integral do sistema ficará indisponível, potencialmente por tempo indeterminado, o que não é ideal. Desta forma, **considera-se o risco desta ameaça como MÉDIO.**

3.3.5 Resumo das ameaças identificadas

Finalizando o levantamento de vulnerabilidades e ameaças, um resumo de todas as ameaças levantadas é exibido na Tabela 3.4.

ID	Ameaça	Categoria	Risco
1	Modificação não autorizada dos dados armazenados	Vazamento de dados pessoais	Alto
2	Apropriação indevida dos dados armazenados via descarregamento de memória do disp.	Vazamento de dados pessoais	Alto
3	Interceptação dos dados via rádio e/ou internet/intranet	Vazamento de dados pessoais	Alto
4	Modificação dos dados por agente externo malicioso intermediário	Vazamento de dados pessoais	Alto
5	Falsificação da autoria dos dados medidos, permitindo o tráfego de dados falsos/inválidos no funcionamento comum do sistema	Vazamento de dados pessoais	Alto
6	Realização de funcionalidades indevidas, como envio de dados medidos para agente externo malicioso ou utilização de recurso computacional para fins diversos não autorizados	Ataques de código malicioso	Médio
7	Identificação não apropriada dos dados de saúde ao paciente correto	Utilização não autorizada de recursos	Médio
8	Utilização do sistema por pessoas não autorizadas, interna ou externamente	Utilização não autorizada de recursos	Médio
9	Negação de serviço intencional	Indisponibilidade de serviço	Médio

Tabela 3.4: Resumo de todos os riscos levantados nesta seção

Vale observar que as ameaças de risco mais alto são aquelas que mais possuem chance de comprometer, com um alto impacto, a privacidade dos dados pessoais sensíveis manejados ao longo do funcionamento do sistema. Isto é proposital, uma vez que, de forma geral, esta pode ser considerada a grande lacuna em termos de segurança da informação no projeto.

Esta tabela é usada como referência na seção seguinte, onde os controles de segurança são sugeridos paras cada tipo de ameaça aqui listado.

3.4 Controles de segurança propostos

Apenas os riscos considerados dentro do nível Alto serão considerados para implementações de segurança, visto que estes são os que comprometem o funcionamento do sistema de forma significativa e possuem chances mais altas de ocorrer. Um fator importante dos riscos analisados que foram classificados como Alto é que todos possuem chances elevadas, direta ou indiretamente, de comprometer os dados sensíveis de saúde dos pacientes envolvidos na utilização da ferramenta, configurando uma situação de emergência na implantação dos controles de segurança.

As recomendações feitas a seguir são direcionadas para cada ameaça listada no resumo da Tabela , sendo referenciadas por AX, onde X representa o ID da ameaça listada na tabela.

3.4.1 A1, A2: Criptografia e assinatura digital dos dados medidos

Com o objetivo de controlar autenticidade, integridade e sigilo dos dados de saúde aferidos, serão implementadas rotinas de criptografia, utilizando AES-128 e assinatura digital, utilizando ECDSA.

A cifragem e assinatura da estrutura que contém os dados de saúde será realizada imediatamente após a medição, de forma que a janela de tempo em que o dados permanece em texto plano é diminuída ao máximo. As rotinas de verificação da assinatura e decifragem ocorrerão no dispositivo *receptor*, de forma que a decifragem ocorrerá apenas em caso de successo na verificação da assinatura digital. Caso contrário, o pacote é simplesmente descartado.

O motivo para a utilização de um método de criptografia simétrica para cifragem junto de uma criptografia assimétrica para assinatura é devido a falta de suporte do kit de desenvolvimento na utilização de criptografia assimétrica para assinatura e cifragem dos dados. Além disso, o ECDSA possui vantagens em relação ao RSA em termos de tamanho de chave, atingindo níveis de segurança equivalentes utilizando menos *bits* para representar seus pares de chaves público e privado(NAKOV S., 2022).

3.4.2 A6: Implementação de assinatura digital no código da aplicação

O código de aplicação e *firmware* dos dispositivos nRF52840 presentes no funcionamento serão assinados digitalmente. A assinatura deve ser verificada no momento de *boot* dos dispositivos, negando o funcionamento usual em caso de falha. As chaves idealmente devem ficar armazenadas em *hardware* seguro

dentro dos microcontroladores. Além disso, toda a operação de cifragem e assinatura dos dados é feita em área segura de memória, descartando risco de vazamento das chaves.

3.4.3 A3, A4, A5: Modificações na comunicação sem fio do sistema

Primeiramente, visando mitigar ataques à nível de WPAN (incluído em A3), pretende-se substituir a utilização do protocolo ESB pelo protocolo Gazell, também proprietário da fabricante e que possui viabilidade de implementação segura (**??**), autenticando e criptografando cada canal de comunicação. Isto dificultará a análise de tráfego de rádio não autorizada, o que por sua controla a integridade, autenticidae e sigilo da comunicação entre os dispositivos nRF52840.

Além disso, o padrão de comunicação WiFi entre o microcontrolador ESP8266, localizado na base centralizadora, e o AP utilizado pela empresa detentora do sistema deve ser WPA-EAP, conhecido também por *enterprise mode*. O padrão oferecido minimiza os riscos das ameaças relacionadas à WiFi e é suportado pelo ESP8266, que realiza a comunicação na LAN/WAN. Caso este padrão não seja utilizado, é extremamente não recomendado que o padrão comunicação não seja o WEP, uma vez que este possui uma série de vulnerabilidades facilmente exploráveis, comprometendo o funcionamento seguro do sistema. Em caso de indisponibilidade do WPA-EAP (*enterprise mode*), a utilização de WPA-PSK deve ser considerada, assumindo os riscos em utilizar uma chave de autenticação compartilhada.

Finalmente, será necessário substituir a utilização do protocolo MQTT pelo MQTT-TLS na comunicação via LAN/WAN com o servidor. O MQTT-TLS é basicamente a implementação do protocolo *Transport Layer Security* sobre o protocolo MQTT, utilizando o conceito de criptografia assimétrica para digitalmente assinar e cifrar a comunicação via internet (DINCULEANă D., 2019), viabilizando a manutenção da privacidade dos dados transmitidos para o servidor.

3.4.4 A6, A7, A8, A9: Riscos assumidos

As ameaças restantes, consideradas de risco médio, terão seus riscos assumidos, uma vez que não apresentam nível considerável de probabilidade que justifique gastos temporais e financeiros para a implementação de controles de segurança.

No entanto, ainda configuram um nível de impacto não ignorável, necessitando de atenção quanto as suas possíveis ocorrências. Portanto, caberá aos mantenedores do SHW analisar e identificar possíveis abusos de uso nestes contextos, mas não se configura urgência quanto implementação dos controles relacionados à estas ameaças.

4
Implementação dos controles de segurança

4.1
Configuração do ambiente de programação

Inicialmente, para a implementação dos controles sugeridos no capítulo anterior, foi necessária a aquisição de 2 dispositivo nRF52840, que é a versão com suporte à *hardware* criptográfico do dispositivo utilizado originalmente.

Em seguida, foi realizado o download da IDE de desenvolvimento *Segger Embedded Studio* e o SDK *nRF52 SDK* fornecido pela *Nordic Semiconductor*. A IDE utilizada é da versão 5.10b, para processadores ARM, enquanto o SDK utilizado está na versão 17.0.2. O código do sistema original foi manualmente entregue pelo autor, de forma que esta foi a única maneira de recuperar o código mais recente.

4.2
Planejamento

A implementação planejada inicialmente consistiu de implementar as modificações diretamente no código original do projeto, começando pelo controle de sigilo, integridade e autenticidade dos dados medidos no dispositivo, seguido pelo estabelecimento de autenticação e criptografia na comunicação de rádio, assinatura digital do *firmware* do dispositivo e finalmente a modificação da comunicação via *internet* para utilizar o protocolo MQTT-TLS ao invés do MQTT simples.

```
<!DOCTYPE CrossStudio_Project_File>
<solution
  Name="nrf_crypto_aes_cbc_with_padding_pca10056"
  target="8"
  version="2">
  <project Name="nrf_crypto_aes_cbc_with_padding_pca10056">
    <configuration
      Name="Common"
      arm_architecture="v7EM"
      arm_core_type="Cortex-M4"
      arm_endian="Little"
      arm_fp_abi="Hard"
      arm_fpu_type="FPv4-SP-D16"
      arm_linker_heap_size="8192"
      arm_linker_process_stack_size="0"
      arm_linker_stack_size="8192"
      arm_linker_treat_warnings_as_errors="No"
      arm_simulator_memory_simulation_parameter="RWX 00000000,00100000,FFFFFFFF;RWX 20000000,00010000,CDCDCDCD"
      arm_target_device_name="nRF52840_xxAA"
      arm_target_interface_type="SWD"
      c_preprocessor_definitions="APP_TIMER_V2;APP_TIMER_V2_RTC1_ENABLED;BOARD_PCA10056;CONFIG_GPIO_AS_PINRESET;DEB
      c_user_include_directories="config;$(SDK)/components;$(SDK)/components/boards;$(SDK)/components/drivers_nrf/n
      debug_register_definition_file="$(SDK)/modules/nrfx/mdk/nrf52840.svd"
      debug_start_from_entry_point_symbol="No"
      debug_target_connection="J-Link"
      gcc_debugging_level="Level 3"
      gcc_entry_point="Reset_Handler"
      linker_output_format="hex"
      linker_printf_fmt_level="long"
      linker_printf_width_precision_supported="Yes"
      linker_scanf_fmt_level="long"
      linker_section_placement_file="flash_placement.xml"
      linker_section_placement_macros="FLASH_PH_START=0x0;FLASH_PH_SIZE=0x100000;RAM_PH_START=0x20000000;RAM_PH_SIZ
      linker_section_placements_segments="FLASH RX 0x0 0x100000;RAM1 RWX 0x20000000 0x40000"
      macros="SDK=/home/rodrigo/.nordic_sdk/nRF5_SDK_17.0.2/nRF5_SDK_17.0.2_d674dde;CMSIS_CONFIG_TOOL=$(SDK)/extern
      project_directory="">
```

Figura 4.1: Amostra do arquivo .emProject, responsável pelas definições de espaços da memória do dispositivo

No entanto, a primeira etapa de implementação foi extremamente complexa e demorada, uma vez que diversos problemas de incompatibilidade de código surgiram durante a implementação. A principal complexidade foi adaptar o arquivo com extensão *emProject* mostrado na Figura 4.1, que é inerente à IDE e vital para a transferência do código compilado ao dispositivo. Neste arquivo são definidas as seções de dados e comandos na memória *flash*, e como estas seções nos dispositivos nRF52832 e nRF52840 são significativamente diferentes, diversas tentativas de execução dos programas compilados resultavam no erro da Figura 4.2. Resolver e entender este erro consumiu muito tempo, originalmente dedicado para implementação, visto que também não existem tantos recursos na *internet* que viabilizassem a descoberta da causa do erro através da pesquisa. Ainda assim, não foi possível solucionar o problema, forçando uma separação de problemas entre portabilidade do código legado para o novo dispositivo e implementação de controles de segurança. Vale ressaltar que a aquisição de um nível de proficiência adequado para o desenvolvimento utilizando o SDK e a IDE citadas exigiu um tempo acima do esperado, o que também contribuiu para uma mudança significativa no planejamento.

```
00> <error> hardfault: HARD FAULT at 0x00000000
00> <error> hardfault:   R0:  0x00000400  R1:  0xFFFFFBFF  R2:  0x200044B4  R3:  0x00000000
00> <error> hardfault:   R12: 0x00000000  LR:  0xFFFFFFF9  PSR: 0x6000003A
00> <error> hardfault: Cause: The processor has attempted to execute an instruction that makes illegal use of the EPSR.
```

Figura 4.2: Output de erro ao executar primeiras modificações no projeto original

Nesse sentido, em função da questão de portabilidade de código entre os diferentes dispositivos, o planejamento inicial foi modificado. Das implementações sugeridas, apenas o controle de integridade, autenticidade e sigilo dos dados no dispositivo medidor foi implementado, em função do curto tempo para implementação restante após as tentativas de adaptar o código original ao novo SoC. Além disso, o código foi produzido em espécie de PoC, com o objetivo de gerar métodos modularizados que sejam aproveitados pelo projeto original quando este for integralmente adaptado ao nRF52840.

Assim, o produto final das implementações é um programa destinado para um dispositivo nRF52840 que realizará o teste da cifragem, assinatura, verificação de assinatura e decifragem de um exemplo de pacote de dados, definido exatamente da mesma forma em que é feito no projeto original. Desta maneira, verificados os resultados das rotinas de cifragem e assinatura, bastará implantar o novo módulo no código original e utilizar as funções criadas para implementar as rotinas.

4.3 Controle de autenticidade e integridade utilizando ECDSA

O *Elliptic Curve Digital Signature Algorithm* é um algoritmo criptográfico assimétrico que se baseia em valores de curvas elípticas na verificação das assinaturas digitais. A motivo da escolha do método se deve à falta de suporte ao método RSA por parte do kit de desenvolvimento, já que este, apesar de mais conhecido e bem estabelcido, principalmente na segurança de comunicações de internet, não apresenta nenhuma API de alto nível para sua utilização. Além disso, uma outra vantagem é que quantidade de *bits* para a representação das chaves públicas e privadas e menor, oferecendo um nível de segurança igual ou maior que o método RSA, que possui chaves maiores (NAKOV S., 2022). Tal fato é vantajoso para dispositivos IoT, uma vez que a quantidade de recursos, como memória principal e secundária, é bem menor, se comparado com computadores pessoais.

Idealmente, a assinatura ocorre no dispositivo medidor, enquanto que a verificação pode ocorrer tanto na base centralizadora quanto após a chegada no servidor MQTT-TLS, quando o dado for processado. Nesse sentido, foi necessário que o par de chaves pública e privada fosse gerado externamente,

importando os *bytes* representativos das chaves para os dispositivos que precisarem. A utilização de cada chave exige a exposição dos *bytes* no código, de forma que não foi possível encontrar informações se a área de memória onde estes *bytes* são consumidos é protegida, representando uma lacuna na implementação. Apesar de não impedir a implementação do esquema de assinatura, essa questão de privacidade das chaves é importante, pois em caso de vazamento da chave privada, toda a rotina de assinatura estará comprometida, uma vez que não será possível discernir dados forjados dos legítimos pela assinatura digital.

A implementação das rotinas de assinatura e verificação é exposta nas Figuras 4.3 e 4.5

```
166  void ecdsa_sign(
167      uint8_t* data,
168      size_t data_size,
169      nrf_crypto_ecdsa_secp256r1_signature_t signature_buffer,
170      size_t* signature_size,
171      uint8_t* raw_private_key,
172      size_t raw_private_key_size
173  )
174  {
175      nrf_crypto_ecc_private_key_t private_key;
176      ret_code_t                   err_code = NRF_SUCCESS;
177
178      NRF_LOG_INFO("Gerando assinatura...");
179
180      // Convert raw private key to internal representation
181      err_code = nrf_crypto_ecc_private_key_from_raw(&g_nrf_crypto_ecc_secp256r1_curve_info,
182                                                     &private_key,
183                                                     raw_private_key,
184                                                     raw_private_key_size);
185      DEMO_ERROR_CHECK(err_code);
186
187      err_code = nrf_crypto_ecdsa_sign(NULL,
188                                       &private_key,
189                                       data,
190                                       data_size,
191                                       signature_buffer,
192                                       signature_size);
193      DEMO_ERROR_CHECK(err_code);
194
195      // Free internal allocations
196      err_code = nrf_crypto_ecc_private_key_free(&private_key);
197      DEMO_ERROR_CHECK(err_code);
198  }
```

Figura 4.3: Implementação da rotina de assinatura digital utilizando ECDSA

```c
void ecdsa_verify(
    uint8_t* data,
    size_t data_size,
    nrf_crypto_ecdsa_secp256r1_signature_t signature_buffer,
    size_t signature_buffer_size,
    uint8_t* raw_public_key,
    size_t raw_public_key_size
)
{
    nrf_crypto_ecc_public_key_t public_key;
    ret_code_t                  err_code = NRF_SUCCESS;

    NRF_LOG_INFO("Verificando assinatura...");

    // Convert raw bytes to public key internal representation
    err_code = nrf_crypto_ecc_public_key_from_raw(&g_nrf_crypto_ecc_secp256r1_curve_info,
                                                  &public_key,
                                                  raw_public_key,
                                                  raw_public_key_size);
    DEMO_ERROR_CHECK(err_code);

    // Data signature verification
    err_code = nrf_crypto_ecdsa_verify(NULL,
                                       &public_key,
                                       data,
                                       data_size,
                                       signature_buffer,
                                       signature_buffer_size);

    if (err_code == NRF_SUCCESS)
    {
        NRF_LOG_INFO("Assinatura valida, o conteudo esta integro e autentico.");
    }
    else if (err_code == NRF_ERROR_CRYPTO_ECDSA_INVALID_SIGNATURE)
    {
        NRF_LOG_WARNING("Assinatura invalida, integridade e autenticidade da mensagem não pode ser verificada");
    }
    else
    {
        // Unpredicted error
        DEMO_ERROR_CHECK(err_code);
    }
}
```

Figura 4.4: Implementação de verificação de assinatura digital utilizando ECDSA

As chaves pública e privada foram geradas utilizando o OpenSSL, renomado utilitário de criptografia e utilizado de forma bem extensiva pela comunidade. A construção do par de chaves foi alcançada conforme demonstrado na Figura 4.5.

```
rodrigo@rm-S145-15IWL:~/tcc/certs$ openssl ecparam -name secp256r1 -genkey -noout -out
 ec-secp256r1-priv-key.pem
using curve name prime256v1 instead of secp256r1
rodrigo@rm-S145-15IWL:~/tcc/certs$ cat ec-secp256r1-priv-key.pem
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIHfFG3bOwLWsdqJomugLOjb/L1vcXoraDwN0LZTMNoq1oAoGCCqGSM49
AwEHoUQDQgAE2suixfmdE4dtPds5ljKKwdqP8pdqW8hS4Su0gods1jTOGkc8Snff
1xExuy7CTx5tzirrmmBuM7OEBGImF0GlaQ==
-----END EC PRIVATE KEY-----
rodrigo@rm-S145-15IWL:~/tcc/certs$ openssl ec -in ec-secp256r1-priv-key.pem -pubout >
ec-secp256r1-pub-key.pem
read EC key
writing EC key
rodrigo@rm-S145-15IWL:~/tcc/certs$ cat ec-secp256r1-pub-key.pem
-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE2suixfmdE4dtPds5ljKKwdqP8pdq
W8hS4Su0gods1jTOGkc8Snff1xExuy7CTx5tzirrmmBuM7OEBGImF0GlaQ==
-----END PUBLIC KEY-----
```

Figura 4.5: Criação do par de chaves para o ECDSA SECP256R1

Idealmente, o par privado da chave deve ser guardado de forma segura

e acessível por todos os dispositivos do sistema que assinarem digitalmente dados antes de uma transmissão. Enquanto isso, o par público não precisa de armazenamento sigiloso, e deve estar disponível para todos os dispositivos que realizarem checagem das assinaturas digitais feitas com seu par privado.

4.4 Controle de sigilo utilizando AES-128

O *Advanced Encryption Standard* é um método criptográfico simétrico, considerado o mais robusto atualmente (KryptAll, 2023). Utilizado no padrão *Cipher-block Chaining*, o método é o mais adequado para manutenção de sigilo dos dados de saúde, visto que os métodos de criptografia assimétrica não estão disponíveis. Portanto, por este ser o melhor dentre os algoritmos simétricos, o AES foi escolhido como método de manutenção de privacidade. A utilização de chaves de 128 *bits* é uma limitação do *hardware* criptográfico presente no nRF52840, que só permite a utilização de chaves deste tamanho para o AES-CBC.

As rotinas de cifragem e decifragem foram implementadas conforme as Figuras 4.6 e 4.7

```
83  void aes_cbc_encrypt(
84      uint8_t* data_in,
85      size_t data_in_size,
86      char* encrypted_buffer,
87      size_t* encrypted_data_size,
88      uint8_t* key
89  )
90  {
91      ret_code_t  ret_val;
92      uint8_t     iv[16];
93      nrf_crypto_aes_context_t cbc_encr_128_ctx;
94
95      /* Init encryption context for 128 bit key and PKCS7 padding mode */
96      ret_val = nrf_crypto_aes_init(&cbc_encr_128_ctx,
97                                    &g_nrf_crypto_aes_cbc_128_pad_pkcs7_info,
98                                    NRF_CRYPTO_ENCRYPT);
99      AES_ERROR_CHECK(ret_val);
100
101     /* Set key for encryption context - only first 128 key bits will be used */
102     ret_val = nrf_crypto_aes_key_set(&cbc_encr_128_ctx, key);
103     AES_ERROR_CHECK(ret_val);
104
105     memset(iv, 0, sizeof(iv));
106     /* Set IV for encryption context */
107
108     ret_val = nrf_crypto_aes_iv_set(&cbc_encr_128_ctx, iv);
109     AES_ERROR_CHECK(ret_val);
110
111     /* Encrypt text
112        When padding is selected m_encrypted_text buffer shall be at least 16 bytes larger
113        than text_len. */
114     ret_val = nrf_crypto_aes_finalize(&cbc_encr_128_ctx,
115                                       data_in,
116                                       data_in_size,
117                                       (uint8_t *)encrypted_buffer,
118                                       encrypted_data_size);
119     AES_ERROR_CHECK(ret_val);
120 }
```

Figura 4.6: Implementação da rotina de cifragem dos dados utilizando AES-128

```
122  void aes_cbc_decrypt(
123      char* encrypted_data,
124      size_t encrypted_data_size,
125      char* decrypted_data,
126      size_t* decrypted_data_size,
127      uint8_t* key
128  )
129  {
130      ret_code_t  ret_val;
131      uint8_t     iv[16];
132      nrf_crypto_aes_context_t cbc_decr_128_ctx; // AES CBC decryption context
133
134
135      /* Init decryption context for 128 bit key and PKCS7 padding mode */
136      ret_val = nrf_crypto_aes_init(&cbc_decr_128_ctx,
137                                    &g_nrf_crypto_aes_cbc_128_pad_pkcs7_info,
138                                    NRF_CRYPTO_DECRYPT);
139      AES_ERROR_CHECK(ret_val);
140
141      /* Set key for decryption context - only first 128 key bits will be used */
142      ret_val = nrf_crypto_aes_key_set(&cbc_decr_128_ctx, key);
143      AES_ERROR_CHECK(ret_val);
144
145
146      /* Set IV for decryption context */
147      memset(iv, 0, sizeof(iv));
148
149      ret_val = nrf_crypto_aes_iv_set(&cbc_decr_128_ctx, iv);
150      AES_ERROR_CHECK(ret_val);
151
152      /* Decrypt text */
153      ret_val = nrf_crypto_aes_finalize(&cbc_decr_128_ctx,
154                                        (uint8_t *)encrypted_data,
155                                        encrypted_data_size,
156                                        (uint8_t *)decrypted_data,
157                                        decrypted_data_size);
158      AES_ERROR_CHECK(ret_val);
159  }
```

Figura 4.7: Implementação da rotina de decifragem dos dados utilizando AES-128

A chave simétrica foi definida de forma aleatória, externamente ao código da aplicação. Isto porque, assim como na geração do par de chaves para a assinatura digital, a decifragem ocorrerá em outros dispositivos, sendo necessário que a chave esteja disponível externamente à eles.

Similarmente ao par de chaves do ECDSA, não há garantias de que a chave simétrica possui a confidencialidade preservada na memória, enquanto é utilizada, representando também uma lacuna desta implementação, uma vez que, em caso de vazamento desta chave, todo o sigilo resultante da utilização do AES será comprometido, o que não é ideal.

5
Resultados e conclusão

A partir da execução do código de testagem das rotinas implementadas, foi possível verificar com sucesso a manutenção do sigilo, integridade e autenticidade de um exemplo de conjunto de dados de exemplo, conforme pode ser observado na Figura 5.1, que demonstram o resultado no terminal da execução do código.

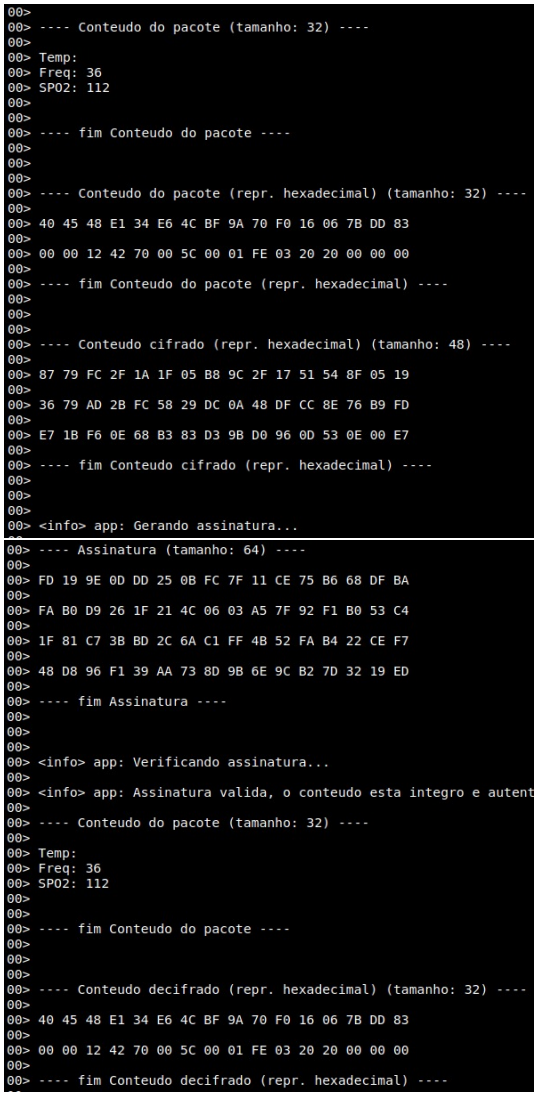

Figura 5.1: Resultado da execução dos testes de cifragem, assinatura, verificação e decifragem de um exemplo de pacote de dados

Um fato importante é que, nas configurações atuais, os *bytes* dos dados estão sendo armazenados em memória no formato *little-endian*, no entanto, o resultado da impressão do dado indica uma possível leitura destes dados em *big-endian*, comprometendo a inteligibilidade dos dados cifrados e assinados.

Ainda assim, como os *bytes* da informação são preservados, este não foi considerado um problema grave da implementação, podendo ser consertado futuramente.

Considerando o resultado final da execução dos testes, é possível concluir que a implementação das rotinas como PoC foi um sucesso para a manutenção do sigilo, integridade e autenticidade dos dados, afirmando-se dessa maneira que, de 4 controles sugeridos, 1 foi construído com sucesso, de forma modularizada. De forma específica, dos 6 objetivos listados na introdução deste relatório, apenas os pontos 5 e 6 não puderam ser verificados integralmente, devido a mudança de planos inesperada.

Apesar da implementação de menos controles do que os sugeridos, a grande vantagem deste projeto foi ter realizado uma extensa Análise de riscos sobre o projeto original, que baseou as escolhas do controles de segurança que não foram implementados. Isto significa que todo o trabalho ainda pode e deve ser aproveitado para a continuação da reformulação, com foco em segurança, deste sistema.

Finalmente, um aprendizado extremamente importante sobre o *redesign* executado é que, quando projetos são desenvolvidos sem a segurança da informação como fundamento, o custo operacional e financeiro para adaptar o código é altíssimo, caracterizando uma situação indesejada em ambientes comerciais, que é o retrabalho. Desta maneira, fica registrada a importância do *security-by-design* na elaboração de sistemas computacionais.

5.1 Trabalhos futuros

Em complemento a este trabalho, devem ser resolvidas as questões de portabilidade do código original, implementado para o nRF52832 e que será utilizando no SoC nRF52840. Além disso, ainda faltam implementar os outros três controles de segurança propostos, que em conjunto com o já implementado neste trabalho, diminuirá os riscos de incidente de segurança da informação do *Smart Health Wearable* à um nível aceitável.

Um outro trabalho que pode ser interessante e complementar a este é a execução de testes de penetração no sistema do SHW como um todo. Um trabalho com este enfoque forneceria um aprendizado extenso sobre segurança ofensiva, especialmente em ambientes IoT. Além disso, seria uma oportunidade de verificar as vulnerabilidades aqui expostas e, potencialmente, encontrar ainda mais vulnerabilidades no sistema, enriquecendo as melhorias de segurança de informação do *Smart Health Wearable*.

6
Referências bibliográficas

BRASIL. **LEI Nº 13.709, DE 14 DE AGOSTO DE 2018**. 2018. Lei Geral de Proteção de Dados Pessoais (LGPD). Brasília, DF: Diário Oficial da União, 2018 [Online; Acessado em 20/07/2023]. Disponível em: <https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm>. Citado na página 9.

COUTINHO, J. **Sistema para monitoramento contínuo, remoto e não-invasivo de pacientes através de dispositivos vestíveis**. 2021. Orientadora: Noemi Rodríguez. 2021. 40 f. Relatório de Projeto Final II - Curso de Graduação em Engenharia da Computação, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2021. Citado 2 vezes nas páginas 12 e 16.

DINCULEANă D., C. X. Vulnerabilities and limitations of mqtt protocol used between iot devices. **Special Issue: Access Control Schemes for Internet of Things**, February 2019. Citado 2 vezes nas páginas 17 e 28.

JAGANNATHAN S., S. A. A cybersecurity risk analysis methodology for medical devices. In: **2015 IEEE Symposium on Product Compliance Engineering (ISPCE)**. [S.l.: s.n.], 2015. p. 1–6. Citado 2 vezes nas páginas 11 e 17.

KryptAll. **HOW SAFE IS AES ENCRYPTION? | ADVANCED ENCRYPTION STANDARD**. 2023. [Online; Acessado em 20/07/2023]. Disponível em: <https://www.kryptall.com/index.php/information/how-safe-is-aes-encryption>. Citado na página 35.

NAKOV S., S. M. S. M. **Practical Cryptography for Developers**. 2022. [Online; Acessado em 20/07/2023]. Disponível em: <https://cryptobook.nakov.com/digital-signatures/ecdsa-sign-verify-messages>. Citado 2 vezes nas páginas 27 e 32.

Nordic Semiconductor. **nRF5 SDK v11.0.0 – Enhanced ShockBurst User Guide**. 2016. [Online; Acessado em 01/10/2022]. Disponível em: <https://developer.nordicsemi.com/nRF_Connect_SDK/doc/latest/nrf/ug_esb.html#>. Citado na página 15.

Nordic Semiconductor. **nRF5 SDK v17.0.2**. 2020. [Online; Acessado em 20/07/2023]. Disponível em: <https://infocenter.nordicsemi.com/index.jsp?topic=%2Fsdk_nrf5_v17.0.2%2Findex.html>. Citado na página 14.

SILVA, A. O. da. **Gestão de Segurança da Informação - Slides**. 2018. Pós-graduação em Segurança da Informação e Comunicações. Cursos de Aperfeiçoamento Avançado para Oficiais - Marinha do Brasil. DI/CETUC/CCEC PUC-Rio. 2018. Citado na página 18.