Aplicação prática da quebra do KeeLoq: uma abordagem de baixo custo

Edison M. Junior, Vinícius Lagrota[1]

[1] Centro de Pesquisa e Desenvolvimento para a Segurança das Comunicações (CEPESC)
Brasília, DF – Brazil.

{junior170287@gmail.com, vinicius.lagrota@gmail.com

Abstract. *This work presents a low-cost attack against the HCS201 integrated circuit, widely used in access control systems based on KeeLoq. Through differential power analysis, it was possible to extract the cryptographic key stored in the device, demonstrating the feasibility of the attack without the need for high-cost measurement equipment. The experiments showed that the manufacturers analyzed use the same cryptographic key across all their devices, making them highly vulnerable to large-scale cloning and piracy attacks. As a possible mitigation strategy, the adoption of key derivation mechanisms, such as HKDF, is proposed to generate unique keys for each transmitter, preventing the extraction of a single key from compromising the entire system.*

Resumo. *Este trabalho apresenta um ataque de baixo custo ao circuito integrado HCS201, amplamente utilizado em sistemas de controle de acesso baseados no KeeLoq. Por meio da análise de potência diferencial, foi possível extrair a chave criptográfica armazenada no dispositivo, demonstrando a viabilidade do ataque sem a necessidade de equipamentos de medição de alto custo. Os experimentos mostraram que os fabricantes analisados utilizam a mesma chave criptográfica em todos os seus dispositivos, tornando-os altamente vulneráveis a ataques de clonagem e pirataria em larga escala. Como possível estratégia de mitigação, é proposto a adoção de mecanismos de derivação de chaves, como o HKDF, para gerar chaves únicas para cada transmissor, impedindo que a extração de uma única chave comprometa todo o sistema.*

1. Introdução

A proteção de informações e a garantia de comunicações seguras tornaram-se aspectos centrais na era digital, impulsionados pela crescente demanda por privacidade, integridade e confiabilidade dos dados em diversos setores [Sun et al. 2025]. Nesse cenário, a criptografia assume papel fundamental ao oferecer mecanismos robustos que asseguram a confidencialidade de informações sensíveis e a autenticidade de transações, tanto em ambientes corporativos quanto em dispositivos pessoais. Com a popularização de sistemas conectados e a expansão da Internet das Coisas (IoT), tornou-se comum a adoção de soluções criptográficas em sistemas embarcados [Thabit et al. 2023]. Esses sistemas, por operarem com recursos computacionais limitados, requerem algoritmos leves, de baixo consumo de energia e com segurança suficiente para proteger dados críticos [Soto-Cruz et al. 2024].

Entretanto, a limitação de recursos e a pressão por custos baixos frequentemente resultam em implementações criptográficas vulneráveis [Ji et al. 2023]. Consequentemente, sistemas embarcados tornam-se alvos comuns de ataques, especialmente os chamados ataques de canal lateral, do inglês *side-channel attack*s (SCAs) [Kocher 1996]. Tais ataques exploram características físicas da execução, como consumo de potência, tempo ou emissão eletromagnética, para recuperar segredos criptográficos. Dentre esses, destaca-se a análise de potência diferencial, do inglês *differential power analysis* (DPA), que aplica técnicas estatísticas sobre traços de energia para inferir valores sensíveis, como chaves criptográficas [Kocher et al. 1999].

Nesse contexto, o algoritmo KeeLoq [Marneweck 1996], desenvolvido para sistemas com restrições de hardware, é amplamente utilizado em controles remotos de portões e sistemas automotivos. Sua popularidade decorre da simplicidade e eficiência, mas, justamente por isso, sua implementação em larga escala torna-se vulnerável à extração de chaves criptográficas. Uma chave comprometida representa ameaça direta à segurança física, permitindo que atacantes obtenham acesso indevido a propriedades privadas, comprometendo o objetivo central de sistemas como os de controle de acesso.

Estudos anteriores demonstraram que é possível quebrar o KeeLoq utilizando SCAs [Paar et al. 2009], mas essas abordagens dependem de equipamentos sofisticados e de alto custo, como osciloscópios de alta taxa de amostragem, dificultando a replicação dos experimentos e limitando seu uso em cenários reais. Isso motiva a busca por alternativas mais acessíveis. Neste trabalho, propõe-se uma abordagem de baixo custo para explorar vulnerabilidades do KeeLoq embarcado nos chips da família HCS da Microchip, utilizando a técnica de DPA. As principais contribuições deste estudo são: i) a descrição da vulnerabilidade explorável no KeeLoq; ii) a implementação prática do ataque com dispositivos de baixo custo; e iii) a discussão de possíveis estratégias de mitigação.

Este artigo está organizado da seguinte forma: a Seção 2 apresenta os trabalhos relacionados; a Seção 3 discute os conceitos necessários para compreensão do estudo; a Seção 4 descreve a análise teórica do ataque; a Seção 5 detalha a implementação prática e discute possíveis mitigações; e a Seção 6 apresenta as conclusões finais.

2. Trabalhos relacionados

Diversos ataques ao KeeLoq podem ser encontrados na literatura. Os primeiros ataques foram publicados por Bogdanov [Bogdanov 2007]. Um dos ataques, baseado na técnica *slide, guess-and-determine* e ataques lineares, precisa de $2^{50.6}$ cifrações do KeeLoq. Já um segundo ataque, mais eficiente, necessita de 2^{37} cifrações. A desvantagem desses ataques é a necessidade de possuir o *codebook* completo, ou seja, ter acesso aos 2^{32} pares de claros-cifrados. Os resultados apresentados por Bogdanov já traziam indícios de severas vulnerabilidades no KeeLoq. Posteriormente, Courtois *et al.* [Courtois et al. 2008] propôs outros ataques ao KeeLoq utilizando somente a técnica *slide*. Em um dos ataques, Courtois *et al.* necessitou de $2^{51.4}$ cifrações do KeeLoq e 2^{16} pares de claros-cifrados. Em seu segundo ataque, este necessitando de quase todo o *codebook*, foi possível revelar a chave secreta usando 2^{27} cifrações. Já Indesteege *et al.* [Indesteege et al. 2008] apresentou um ataque prático ao KeeLoq, no qual é capaz de recuperar a chave secreta com complexidade de $2^{44.5}$ cifrações utilizando ataques de *slide* e *meet-in-the-middle*. Em um ataque, usando poucos pares de claro-cifrado do *codebook*, a chave secreta pode ser re-

cuperada em menos de 8 dias em CPUs com 64 núcleos. Já usando um *codebook* de 2^{16} pares de claro-cifrado, é capaz de executar o ataque em menos de 4 dias no mesmo *setup*.

Apesar de avanços consideráveis, os ataques citados são aplicados somente ao KeeLoq identification friend or foe (IFF), mas não são aplicáveis ao modo *Code Hopping*. De forma resumida, o IFF é um sistema de autenticação bidirecional, onde um dispositivo envia um desafio criptográfico e aguarda uma resposta única gerada a partir de uma chave secreta, garantindo maior segurança contra ataques de repetição. Já o modo *Code Hopping* é um mecanismo de autenticação unidirecional, onde cada transmissão usa um código variável gerado por um algoritmo, impedindo ataques de captura e repetição de códigos, sendo amplamente utilizado em controles remotos de veículos e portões. Neste sentido, devido à impossibilidade de ataques de captura e repetição, os ataques acima ficam inviáveis devido à impossibilidade de construir um *codebook*, já que o adversário terá acesso apenas ao cifrado, neste caso. Tendo em vista que os produtos comerciais utilizam o KeeLoq no modo *Code Hopping*, como, por exemplo, os chips da família HCS da Microchip, os ataques acima não são possíveis de serem aplicados.

No entanto, [Eisenbarth et al. 2008] propôs um ataque ao KeeLoq no modo *Code Hopping* em chips da famíla HCS utilizando SCA, mais especificamente, DPA. Combinando DPA com características específicas do KeeLoq, foi possível revelar de forma eficiente a chave secreta armazenada no dispositivo. Ainda foi mostrado como deve ser o procedimento para clonar um dispositivo e como realizar uma negação de serviço (DoS) em sistemas de controle de acesso com usam o KeeLoq. Apesar do trabalho de Eisenbarth *et al.* quebrar completamente o KeeLoq, foram necessário a utilização de equipamentos de alto custo e de difícil mobilidade. Neste sentido, uma lacuna notável na literatura existente diz respeito a soluções igualmente eficazes à apresentada por Eisenbarth *et al.*, mas que necessitam de um custo significantemente menor do que o requerido no estudo mencionado. Portanto, o presente trabalho busca preencher esta lacuna apresentando um método de baixo custo para a quebra do KeeLoq.

3. Fundamentos

Esta seção apresenta de forma detalhada o Rolling Code, técnica utilizada no KeeLoq. Além disso, os principais aspectos do KeeLoq são discutidos.

3.1. A tecnologia Rolling Code

A tecnologia Rolling Code (código rotativo), é usada principalmente em dispositivos de controle remoto, como alarmes automotivos e portões eletrônicos, trazendo maior nível de segurança por evitar que códigos de acesso sejam interceptados e reutilizados por terceiros. Essa tecnologia baseia-se em criptografia simétrica e é fundamentalmente simples uma vez que é utilizada em dispositivos de baixo poder de processamento e de baixo custo. No funcionamento dessa tecnologia apenas um dispositivo transmite (o transmissor) e apenas o outro dispositivo recebe (receptor) a informação. Cada vez que o transmissor efetua uma transmissão, ele gera um código sequencial que é transmitido de forma criptografada. Uma vez que esse código é recebido pelo receptor, este tem a capacidade de avaliar se tal código é válido baseando no código anteriormente transmitido. Se um código antigo for interceptado e utilizado novamente, ele será automaticamente rejeitado, uma vez que cada código só é aceito uma única vez.

3.1.1. O algoritmo KeeLoq

O KeeLoq é um algoritmo criptográfico simétrico com chave de 64 bits e bloco de 32 bits, desenvolvido para a tecnologia Rolling Code, garantindo que cada transmissão entre o transmissor e o receptor seja única e segura, protegendo contra ataques de clonagem ou interceptação. Esse mecanismo é amplamente utilizado em sistemas de controle de acesso, como anti-furto veicular e abridores de portas de garagem. No funcionamento do sistema, o transmissor contém um codificador e o receptor, um decodificador, ambos compartilhando uma chave secreta. Além disso, a comunicação é sincronizada por um contador de 16 bits, que é incrementado a cada nova transmissão do código variável [Eisenbarth et al. 2008].

A função de cifração do KeeLoq, ilustrada na Figura 1, pode ser representada matematicamente como uma permutação bijetiva sobre um espaço de blocos de 32 bits $E_K : \{0,1\}^{32} \rightarrow \{0,1\}^{32}$, tal que E_K representa a função de cifração parametrizada por uma chave secreta K de 64 bits. Isso significa que E_K recebe uma entrada P (um bloco de 32 bits) e produz uma saída C (outro bloco de 32 bits), garantindo que cada entrada tenha exatamente uma saída correspondente (função bijetiva): $C = E_K(P)$, com $P, C \in \{0,1\}^{32}$.

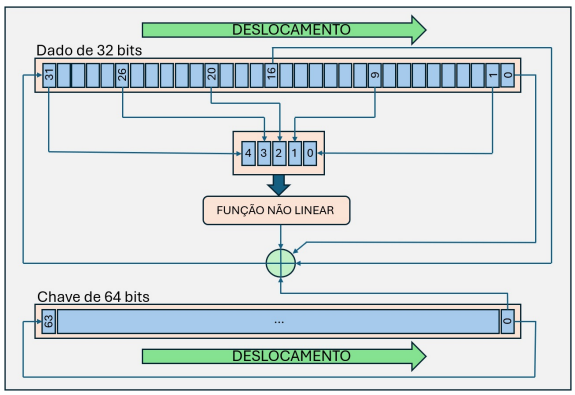

Figura 1. Função criptográfica de cifração do KeeLoq.

Após a inserção da chave e do valor em claro, são executados 528 ciclos de processamento, ao final dos quais o bloco de dados conterá o valor cifrado. Em cada ciclo de execução, tanto o barramento de dados quanto a chave são deslocados em um bit para a direita. Simultaneamente, os bits de posição 31, 26, 20, 9 e 1 do bloco de dados são utilizados como entrada para a função não-linear, do inglês *non-linear function* (NLF), que produz um único bit de saída. No mesmo ciclo, os bits de posição 16 e 0 do bloco de dados, combinados com o bit de saída da NLF e o bit 0 da chave, são processados por meio de uma operação XOR. O resultado dessa operação determina o bit que será inserido na posição 31 do barramento de dados após o deslocamento. Além disso, o bit 0 da chave é realocado para a posição 63 após o deslocamento, criando um efeito de rotação da chave a cada ciclo.

O processo de decifração segue a lógica inversa da cifração. Os 32 bits do bloco cifrado são inseridos na posição correspondente aos dados, e a mesma chave de 64 bits utilizada na cifração é carregada em sua posição original. Para recuperar os dados originais,

todas as operações realizadas na cifração são revertidas, aplicando as funções opostas em cada etapa, resultando na restauração do bloco de dados original.

3.2. O circuito integrado HCS201

O circuito integrado HCS201 [Microchip 2001] faz parte de uma linha de chips desenvolvidos pela Microchip Technology e é amplamente utilizado em sistemas de controle de acesso e segurança. Esse chip disponibiliza uma implementação da tecnologia Rolling Code e faz uso do algoritmo KeeLoq. Além disso, ele é adequado para aplicações de baixo custo, pois a criptografia é realizada diretamente pelo chip, além de gerar um sinal de saída próprio para ser aplicado a um transmissor de radiofrequência.

O chip contém internamente uma chave de 64 bits, um número serial de 28 bits e um contador de 16 bits. Inicialmente, o chip é fabricado sem a chave e o número serial, permitindo que o fabricante do dispositivo criptográfico grave essas informações durante a configuração. Para isso, o chip dispõe de uma memória não-volátil do tipo EEPROM, onde são armazenados o número serial, a chave e o valor do contador, que é incrementado a cada acionamento. Uma vez gravada, a chave é utilizada internamente no processo criptográfico, mas não pode ser lida externamente por nenhuma interface ou programa – sendo acessível apenas ao próprio circuito interno do chip durante a execução das operações criptográficas.

3.2.1. Montagem do pacote de dados

O circuito integrado HCS201 monta um pacote de dados de 66 bits, conforme ilustrado na Figura 2. Os primeiros 34 bits são transmitidos em claro e os 32 bits finais correspondem a dados cifrados com o algoritmo KeeLoq.

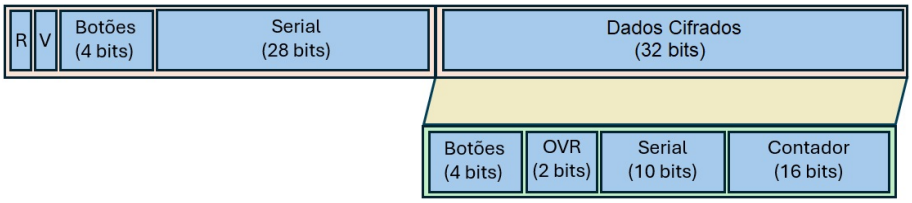

Figura 2. Pacote de dados de 66 bits montados pelo HCS201.

Os bits em claro incluem: i) R (1 bit), que indica se é a primeira transmissão do pacote (valor 0) ou uma retransmissão contínua enquanto o dispositivo permanece acionado (valor 1); ii) V (1 bit), que sinaliza baixa tensão de alimentação, útil para alertas em dispositivos com bateria; iii) *Botões* (4 bits), que indicam quais terminais do chip foram acionados, permitindo até 16 combinações distintas; e iv) *Serial* (28 bits), que representa o número serial único do chip, utilizado para identificação pelo receptor. Com 28 bits, é possível endereçar mais de 250 milhões de dispositivos.

Os 32 bits finais do pacote são cifrados e somente podem ser decifrados por quem possui a chave criptográfica correspondente. Estes incluem: i) *Bits dos botões* (4 bits), que duplicam a informação dos botões transmitida em claro; ii) *Bits* OVR (2 bits), que são incrementados quando o contador atinge seu limite e permanecem em 11 (binário) após isso; iii) *Dado serial* (10 bits), correspondentes aos 10 bits menos significativos do

número serial, usados na validação da decifração; e iv) *Contador* (16 bits), incrementado a cada acionamento para garantir a sequência e sincronização com o receptor.

A sincronização do contador é essencial para garantir a autenticidade da comunicação. Como pode ocorrer de o transmissor ser acionado sem que o receptor receba o sinal, o contador do transmissor pode avançar sem que o receptor acompanhe. Caso o receptor receba um valor com diferença de 1 a 16 em relação ao último valor válido, a sincronização é automática. Se a diferença for entre 17 e 128, são exigidas duas transmissões consecutivas válidas. Por fim, para diferenças acima de 128, o número serial é removido da lista de dispositivos confiáveis.

Essa última medida previne ataques em que um invasor tente transmitir pacotes de dados forjados com os 32 bits criptografados variando a cada transmissão, na tentativa de gerar aleatoriamente um código válido. Como a probabilidade de gerar um dos 16 valores aceitáveis ou dois valores consecutivos dentre os 112 esperados sem gerar um valor inválido é extremamente baixa, essa estratégia reforça a segurança do sistema contra esse tipo de ataque.

4. Criptanálise do circuito integrado HCS201

Nesta seção será apresentada de forma detalha a vulnerabilidade do KeeLoq e como é possível explorá-las no HCS201 usando DPA. Conforme mostrado por [Eisenbarth et al. 2008], a vulnerabilidade da implementação do algoritmo baseia-se no princípio de que o algoritmo Keeloq é implementado em hardware no HCS201 e que os circuitos de memória interna dissipam mais potencia elétrica quando há variação do estado lógico de seus bits. Como o circuito integrado em questão envia os códigos que são obtidos no final do processo criptográfico (para o acionamento do receptor), o algoritmo Keeloq pode ser atacado de trás pra frente, ou seja, começando no último ciclo executado.

A Figura 3 mostra o estado dos registradores de dados (os mesmos 32 bits de dados cifrados apresentados na Figura 2) e da chave nos três últimos ciclos do algoritmo. Os 32 bits de dados, denotados por $\mathbf{B} = (b_0, b_1, ..., b_{31})$, são conhecidos, já que são os dados presentes no registrador após todo o processo criptográfico e assim, são transmitidos pelo dispositivo. Os bits da chave, denotados por $\mathbf{K} = (k_0, k_1, ..., k_{63})$, assim como os bits denominados x_0 e x_1 na Figura 3 não são inicialmente conhecidos, pois são valores intermediários de b_0 nos ciclos 527 e 526, respectivamente. É a partir deles que será feito o ataque. Com base na descrição do KeeLoq feita na Seção 3.1.1, temos as seguintes relações:

$$\begin{aligned} b_{31} &= b_{15} \oplus NLF\left(b_{30}, b_{25}, b_{19}, b_8, b_0\right) \oplus k_{15} \oplus x_0 \\ b_{30} &= b_{14} \oplus NLF\left(b_{29}, b_{24}, b_{18}, b_7, x_0\right) \oplus k_{14} \oplus x_1 \end{aligned} \tag{1}$$

Rearranjando a Equação 1, pode-se isolar os bits x_0 e x_1:

$$\begin{aligned} x_0 &= b_{15} \oplus NLF\left(b_{30}, b_{25}, b_{19}, b_8, b_0\right) \oplus k_{15} \oplus b_{31} \\ x_1 &= b_{14} \oplus NLF\left(b_{29}, b_{24}, b_{18}, b_7, x_0\right) \oplus k_{14} \oplus b_{30} \end{aligned} \tag{2}$$

É possível perceber que o conhecimento do bit x_0 está diretamente relacionado ao bit k_{15} da chave, pois todos os outros valores relacionados são conhecidos. Sendo assim, é possível calcular o valor do bit x_0 supondo um valor para o bit k_{15} da chave. Partindo disso, é possível avaliar as duas suposições de $k_{15} \in \{0, 1\}$ e para cada uma delas classificar a amostra dos dados conhecidos em dois subconjuntos:

1. $x_0 = b_0$: o primeiro bit do registrador não sofreu mudança do estado lógico na mudança do ciclo 527 para o ciclo 528.
2. $x_0 \neq b_0$: o primeiro bit do registrador sofreu mudança do estado lógico na mudança do ciclo 527 para o ciclo 528.

A Figura 4 mostra de maneira simplificada a divisão do conjunto de todos os traços obtidos pelos acionamentos do HCS201 em quatro subconjuntos. Cada par de subconjunto é resultado das suposições ($k_{15} = 1$ e $k_{15} = 0$). Cada par se subdividem em função da igualdade ou diferença entre x_0 e b_0. Para simplificar, são apresentadas apenas 10 amostras nomeadas de A até J.

Analisando a divisão do conjunto de amostras, é possível perceber que os dois pares de subconjuntos formados por cada suposição do bit da chave ($k_{15} = 1$ e $k_{15} = 0$) são iguais mudando apenas a ordem de acordo com a igualdade ou diferença entre x_0 e b_0. Isso ocorre porque a relação do bit x_0 com o bit k_{15} da chave é linear. Neste sentido, é necessário escolher um valor que tenha um alto grau de não linearidade em relação à chave [Eisenbarth et al. 2008].

Como as duas suposições para o bit k_{15} da chave são simétricas, torna-se impossível avaliar qual é a suposição correta. Já o conhecimento do bit x_1 está diretamente relacionado ao bit k_{14} da chave e ao bit x_0. Como o bit x_0 está relacionado ao bit k_{15} da chave, o bit x_1 também está relacionado com o bit k_{15} da chave como dado de entrada da NLF. Isso pode ser demonstrado rearranjando a Equação 2:

$$x_1 = b_{14} \oplus NLF\left(b_{29}, b_{24}, b_{18}, b_7, (b_{15} \oplus NLF\left(b_{30}, b_{25}, b_{19}, b_8, b_0\right) \oplus k_{15} \oplus b_{31})\right) \oplus k_{14} \oplus b_{30} \quad (3)$$

Ainda, o conhecimento do bit x_1 está diretamente relacionado aos bits k_{15} e k_{14} da chave, pois todos os outros valores relacionados são conhecidos. Sendo assim, é possível

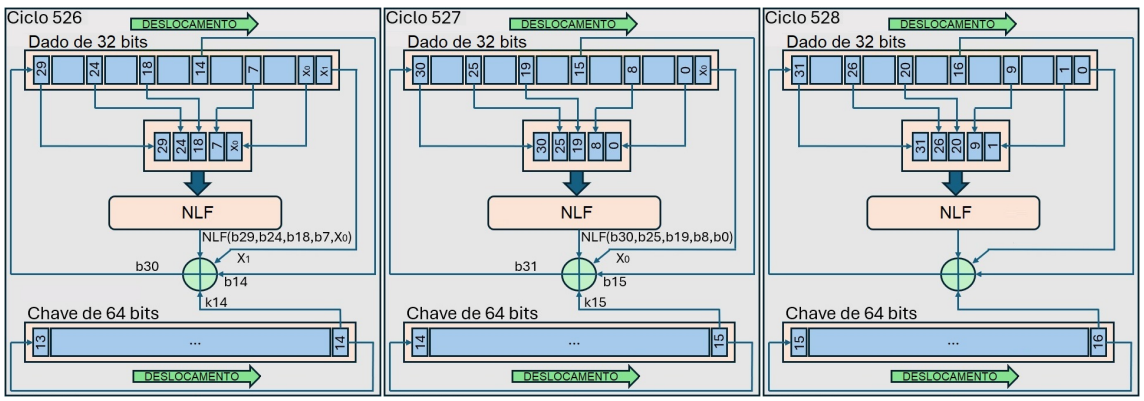

Figura 3. Estado dos registradores.

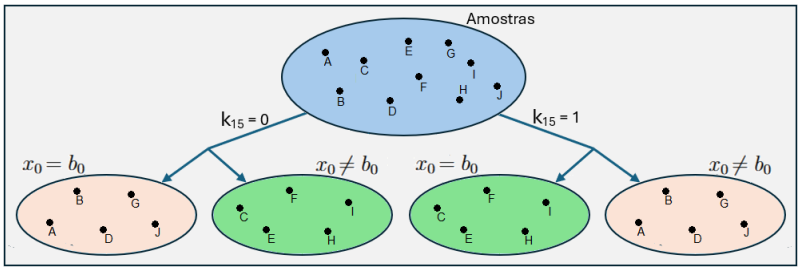

Figura 4. Divisão das amostras em função de x_0 e b_0.

calcular o valor do bit x_1 supondo um valor para o bit k_{15} da chave e um valor para o bit k_{14} da chave. No entanto, como o bit k_{14} da chave está relacionado ao bit x_1 de forma linear, define-se qualquer valor para ele e avalia-se as duas suposições de $k_{15} \in \{0, 1\}$ e para cada uma delas classifica-se a amostra dos dados conhecidos em dois tipos:

1. $x_1 = x_0$: o primeiro bit do registrador não sofreu mudança do estado lógico na mudança do ciclo 526 para o ciclo 527.
2. $x_1 \neq x_0$: o primeiro bit do registrador sofreu mudança do estado lógico na mudança do ciclo 526 para o ciclo 527.

A Figura 5 mostra de maneira simplificada a divisão do conjunto de todos os traços obtidos pelos acionamentos do HCS201 em quatro subconjuntos. Cada par de subconjunto é resultado das suposições ($k_{15} = 1$ e $k_{15} = 0$). Cada par se subdividem em função da igualdade ou diferença entre x_1 e x_0. Para simplificar, são apresentadas apenas 10 amostras nomeadas de A até J. Além dos dados $\mathbf{B} = (b_0, b_1, ..., b_{31})$, cada uma das amostras contem o traço de potência emitido pelo HCS201 durante o seu respectivo acionamento.

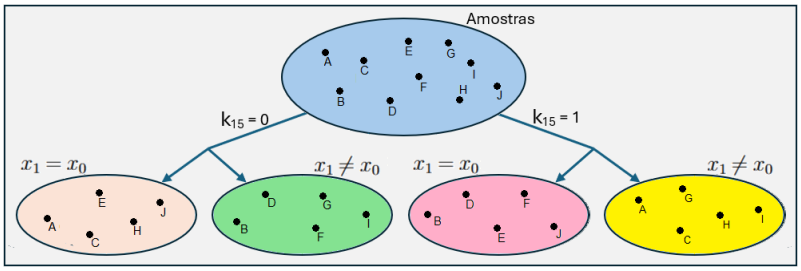

Figura 5. Divisão das amostras em função de x_1 e x_0.

Analisando a divisão do conjunto de amostras é possível perceber que os dois pares de subconjuntos gerados são formados de forma totalmente diferente para cada suposição de valor do bit k_{15} da chave. Isso ocorre porque a relação do bit x_1 com o bit k_{15} da chave não é linear pois o bit da chave k_{15} se relaciona com o bit x_1 como um dado de entrada da função NLF. Como o bit k_{14} da chave está relacionado com o bit x_1 de forma linear, a mudança da suposição de seu valor apenas irá inverter a formação dos pares de subconjuntos, tornando a suposição de seu valor não relevante.

Como os dois pares de subconjunto gerados são totalmente diferentes para cada suposição do valor do bit k_{15} da chave, através de DPA é possível explorar a correlação entre o consumo de potência e a manipulação de bits internos do dispositivo, já que apenas

a suposição correta do bit k_{15} da chave apresentará dois subconjuntos de amostras de dados de saída cujos consumos de potência se diferem significativamente. Isso ocorre porque apenas para a suposição correta o subconjunto $x_0 \neq x_1$ será formado por amostras cujo consumo de potência será mais alto durante os 31 últimos ciclos em que estes dados deslocam pelo barramento de dados já que nestes 31 ciclos o bit de memória que contém x_0 passará a conter x_1, ou seja, uma variação de estado lógico.

Embora os outros bits de dados também apresentem variações de estado lógico e, consequentemente, consumo de potência, mascarando o consumo de potência que revela o bit k_{15} da chave, estatisticamente, se for usada uma quantidade de amostras suficientemente grande, estas variações de potência serão distribuídas por igual nos dois subconjuntos. Sendo assim, basta calcular a média do consumo de potência referente aos dois subconjuntos separadamente e compará-los. Os dados referentes ao consumo de potência de outros bits e até mesmo de eventuais ruídos tenderão a se anularem sobrando apenas o consumo de potência referente ao bit k_{15} da chave. Como a diferença do consumo de potência só existirá se os dois subconjuntos tiverem de fato valores de consumo de potência distintos, analisando os traços da diferença dos consumos de potência para as duas suposições de valores para o bit k_{15} da chave, apenas um dos traços irá apresentar um pico positivo ou negativo que se destacará do valor médio de todo o traço e assim revela-se o valor correto do bit k_{15} da chave

Uma vez conhecido o bit k_{15} da chave, pode-se conhecer o valor do bit x_0 e todos os estados lógicos dos registradores durante o ciclo 527. Sendo assim, basta repetir a análise, só que ao invés de analisar os ciclos 526 e 527, analisa-se os ciclos 525 e 526, pois assim é possível obter o bit k_{14} da chave. Com o bit k_{14} da chave revelado, executa-se o processo de análise dos dados nos ciclos anteriores e obtém-se o bit k_{13}, que por sua vez possibilita repetir o processo para obter o bit k_{12} e assim sucessivamente até o último bit da chave ser revelado. O processo de análise dos dados precisa ser repetido para cada bit da chave. No entanto, isso não se apresenta como um problema uma vez que essa análise é rápida e fácil de ser processada.

5. Implementação do ataque

Nesta seção será apresentado todo o processo realizado para a quebra da chave criptográfica do circuito integrado HCS201 descrevendo as fases de coleta de dados e do processo de DPA para a recuperação da chave secreta.

Para realizar tais medições, [Eisenbarth et al. 2008] utilizaram um osciloscópio Agilent 54832D, com taxa de amostragem de até 4 GS/s e capacidade de armazenamento de 128 Mpts, o que possibilita a captura de sinais de alta frequência com elevada resolução temporal e durante janelas de tempo relativamente longas. Graças a essa taxa de amostragem elevada, os dados obtidos apresentaram excelente fidelidade, permitindo a extração da chave criptográfica com apenas algumas dezenas de traços. Essa alta precisão reduziu significativamente a necessidade de coleta e processamento de grandes volumes de dados.

Em contraste, nossa proposta consistiu na construção de um circuito dedicado para aquisição de sinais, com custo extremamente reduzido. Esse circuito foi capaz de operar com uma taxa de amostragem de apenas 250 kS/s, muito inferior àquela oferecida por osciloscópios profissionais. Embora tenha sido possível armazenar um número razoável de amostras, a baixa resolução temporal e a ausência de um sistema de trigger preciso

tornaram os dados coletados mais ruidosos e imprecisos. Como consequência, foi necessário coletar aproximadamente 10 mil traços, além de implementar etapas adicionais de processamento, como alinhamento por picos de consumo e filtragem de ruído, a fim de viabilizar a extração da chave criptográfica. A Tabela 5 apresenta uma comparação entre as características do trabalho de Eisenbarth *et al.* e as características deste trabalho.

Tabela 1. Comparação entre [Eisenbarth et al. 2008] e este trabalho.

	[Eisenbarth et al. 2008]	Este trabalho
Hardware de aquisição	Agilent 54832D	Circuito próprio
Taxa de amostragem	Até 4 GS/s	250 kS/s
Precisão das amostras	Alta	Baixa
Volume necessário de dados	Baixo	Alto
Capacidade de armazenamento	128 Mpts	500 kpts
Custo	Acima de US$ 1.000,00	Abaixo de US$ 25,00

Embora a técnica de exploração do bit x_0, baseada na Equação 2, derive de observações originalmente apresentadas por Eisenbarth *et al.*, o modelo estatístico empregado em nosso trabalho foi significativamente simplificado para operar com hipóteses binárias diretas (mudança ou não de estado lógico). Isso permitiu uma análise mais compatível com o uso de hardware acessível e limitado. Assim, não utilizamos o CPA tradicional, que exige a definição de modelos de vazamento (como Hamming Weight ou Hamming Distance) e a extração precisa de picos. Em vez disso, agrupamos os traços com base na previsão de transição de bits, estratégia mais robusta frente ao ruído e à baixa resolução dos traços.

Dessa forma, ainda que inspirado na abordagem anterior, este trabalho é inédito ao demonstrar que esse tipo de ataque pode ser realizado com sucesso utilizando apenas equipamentos de baixo custo, o que amplia significativamente seu impacto prático. Ao eliminar a barreira econômica associada à instrumentação sofisticada, a ameaça ao KeeLoq se torna ainda mais crítica, viabilizando ataques reais em cenários antes considerados inviáveis, como discutido na Seção 4.

5.1. Descrição do hardware de baixo custo

Para este trabalho, o STM32H743ZI foi escolhido devido à alta frequência operacional de sua CPU (500 MHz), à disponibilidade suficiente de memória RAM (564 kB) para armazenar as amostras coletadas, e à presença de um conversor analógico-digital (ADC) integrado de 8 bits, além do seu baixo valor de mercado, abaixo de US$ 20,00. Com essa configuração, é possível obter uma taxa de amostragem de aproximadamente 250 kS/s, o que equivale a 4 μs entre amostras. Portanto, a taxa de amostragem do ADC do STM32H743ZI é relativamente baixa, algo comum em dispositivos de baixo custo, especialmente se comparada as taxas obtidas por equipamentos dedicados de medição, como osciloscópios.

A Figura 6 apresenta o hardware de baixo custo desenvolvido. O circuito foi construído de forma modular, permitindo que outros chips, não somente aqueles da família

HCS, também possam ser analisados futuramente. A placa à esquerda abriga o chip HCS201, alvo da extração da chave criptográfica, além de alguns *jumpers* utilizados para selecionar qual porta será acionada durante a captura dos dados. A placa central contém o circuito integrado OPA4354A, um amplificador operacional de alta velocidade, responsável por amplificar os sinais de consumo de corrente do dispositivo. Já a placa à direita contém o módulo STM32H743ZI, anteriormente mencionado, que realiza a amostragem e transmissão dos dados. Esses módulos estão interligados por um cabo *flat*, que conecta dois periféricos adicionais: um conversor USB/Serial e um gravador ST-LINK V2. O conversor USB/Serial estabelece a comunicação entre o microcontrolador e o computador responsável pela coleta e armazenamento dos dados. O gravador ST-LINK V2, por sua vez, é utilizado tanto para carregar o *firmware* no microcontrolador quanto para fornecer alimentação elétrica ao circuito durante sua operação.

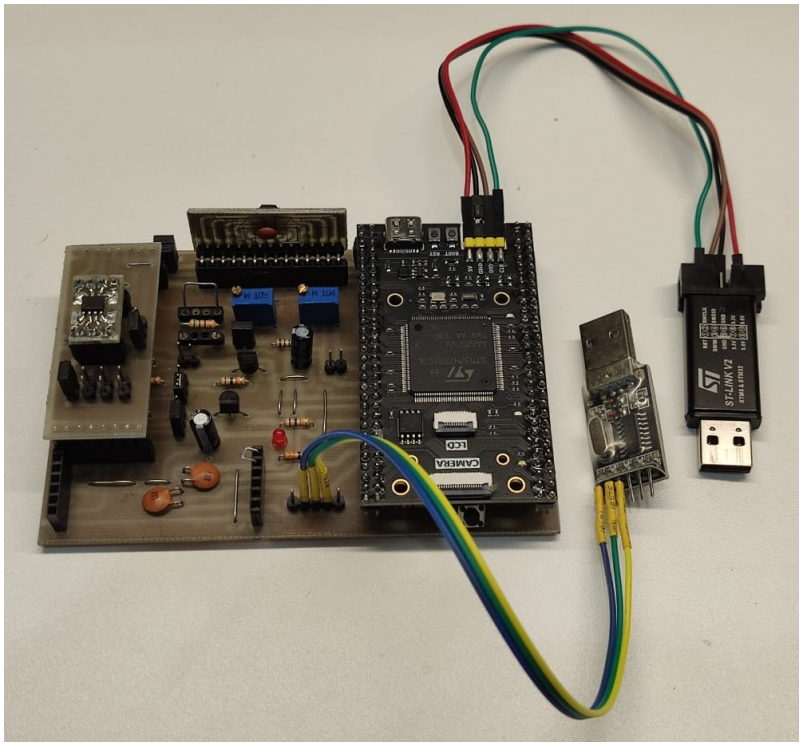

Figura 6. Hardware de baixo custo construido com PCB.

5.2. Coleta de dados

Para que o DPA seja aplicado, é necessário medir o consumo de potência do circuito integrado HCS201 com precisão. Para isso, foi utilizado um resistor conectado ao terminal negativo de alimentação do dispositivo, garantindo que toda a corrente consumida pelo HCS201 passe pelo resistor. A corrente consumida gera uma pequena queda de tensão no resistor, que é proporcional ao consumo de potência e pode ser amplificada e medida diretamente. A captura dessa diferença de potencial foi feita utilizando um amplificador operacional de baixo ruído, elevando o sinal para um nível adequado ao ADC integrado no STM32H743ZI.

Para compensar a limitação da taxa de amostragem do STM32H743ZI, foi necessário coletar um volume maior de dados, permitindo minimizar a imprecisão das

medições individuais através da média estatística das múltiplas leituras realizadas. Portanto, para a coleta de dados, a cada 2 segundos o próprio microcontrolador aciona o HCS201, faz a coleta de 6 mil amostras do consumo de potência, o que equivale a um intervalo de 24 ms, e armazena em sua memória RAM interna. Em seguida, efetua a leitura do sinal elétrico emitido pelo HCS201 para obter os dados calculados após o processo criptográfico (pacote de dados de 66 bits ilustrado na Figura 2). Por fim, envia os dados para um computador através de sua porta serial. No total, foram realizados 10 mil acionamentos do HCS201, cada acionamento constituído por 6 mil amostras do consumo de potência. Sendo assim, a fase de coleta levou menos de 6 horas.

5.3. Análise diferencial de potência

A Figura 7 apresenta o gráfico do consumo de potência do circuito integrado HCS201 durante um acionamento, abrangendo um conjunto de 6 mil amostras adquiridas pelo ADC.

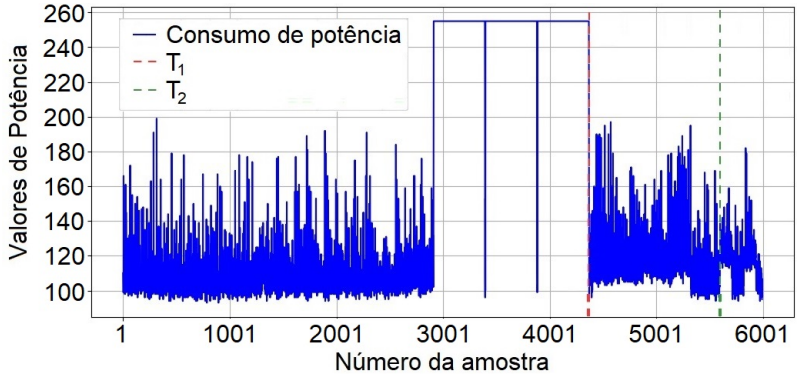

Figura 7. Consumo de potência de um acionamento do HCS201.

Observa-se que o consumo de potência medido varia entre os valores digitais aproximados de 95 e 200. Nota-se claramente, no entanto, que no meio do intervalo analisado há um aumento expressivo no consumo, chegando a saturar a medição do ADC. Esse alto consumo de potência ocorre durante o processo de leitura e escrita na EEPROM interna do HCS201 [Eisenbarth et al. 2008]. Dado que o processo criptográfico deve ocorrer após o acesso à EEPROM, tendo que precisa-se acessar o número serial, a chave e o número de contador armazenado nela, pode-se supor que a operação de criptografia ocorre imediatamente após essa elevação abrupta do consumo de potência, identificado no gráfico como T_1. Além disso, no ponto identificado no gráfico como T_2, observa-se outra variação perceptível no consumo, que coincide com a geração do sinal de saída no pino correspondente do circuito integrado. Esta região final representa, portanto, o consumo decorrente da transmissão efetiva do sinal criptografado (pacote de dados). Tal período encontra-se claramente diferenciado e ocorre após a região de criptografia mencionada anteriormente. Essas observações permitem concluir que o trecho do gráfico mais adequado para análise via DPA está situado logo após o pico de consumo de potência (T_1) e antes do início da transmissão do sinal criptografado (T_2).

A Figura 8 apresenta o gráfico do consumo de potência de dois acionamentos distintos do circuito integrado HCS201, abrangendo o período que se inicia pouco antes do término do acesso à memória EEPROM e se estende até logo após o início da transmissão

do sinal de saída. Nota-se que os traços de consumo apresentam certa dessincronização, o que é esperado, uma vez que o HCS201 não possui um oscilador interno de alta precisão para garantir tempos de execução idênticos entre acionamentos consecutivos. Para viabilizar a aplicação da DPA e a extração da chave criptográfica, é essencial sincronizar todas as 10 mil amostras de dados coletadas. Esse alinhamento é realizado tomando como referência dois pontos característicos no gráfico: a queda brusca no consumo de potência logo após o acesso à EEPROM (T_1) e o aumento significativo de consumo de potência no início da transmissão do sinal de saída (T_2). Esses momentos foram escolhidos por serem facilmente identificáveis, permitindo um realinhamento preciso dos traços e garantindo a eficácia da análise.

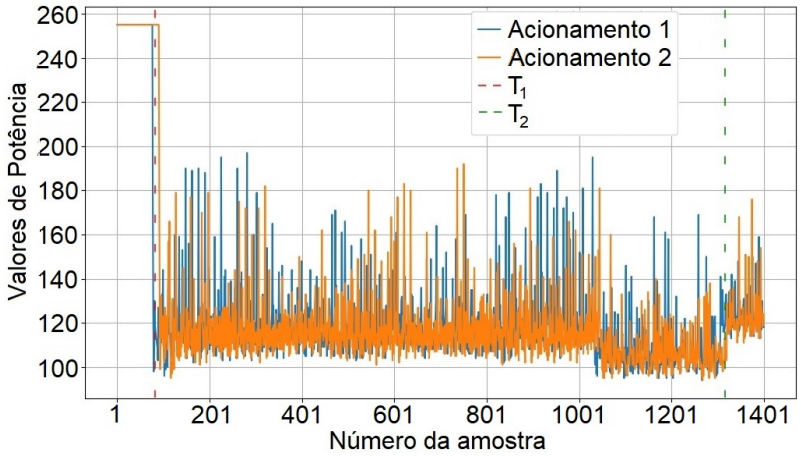

Figura 8. Consumo de potência de dois acionamentos do HCS201.

Para a sincronização, todas as amostras anteriores a T_1 e posteriores a T_2 foram removidas, assegurando que os traços dos 10 mil acionamentos iniciem exatamente em T_1 e terminem em T_2. No entanto, alguns traços possuem um número maior de amostras que outros devido à imprecisões do HCS201. Para padronizar a quantidade de amostras em todos os traços, foram inseridas amostras adicionais nos traços com menor quantidade de pontos, preservando as características temporais e a coerência dos dados. O processo de ajuste ocorre em etapas. Primeiramente, determina-se o traço com o maior número de amostras, que será utilizado como referência. Em seguida, cada traço com menor quantidade de amostras é ajustado para igualar sua contagem à do traço de referência. O número total de amostras a serem adicionadas é calculado e distribuído uniformemente ao longo do traço. Por exemplo, se for necessário inserir duas amostras adicionais, o traço é dividido em três segmentos de tamanho aproximadamente igual. Nos pontos de inserção, calcula-se a média aritmética entre os valores digitalizados das amostras adjacentes, garantindo que as novas amostras sejam inseridas de forma suave e consistente, minimizando distorções no sinal original. Além disso, passou-se um filtro passa-baixa média móvel em cada um dos traços de potência, removendo componentes de alta frequência que não contribuem para a análise e atrapalham na detecção. Após a aplicação do filtro passa-baixa, aplica-se a DPA baseando-se na vulnerabilidade do KeeLoq discutido na Seção 4.

Inicialmente, supõe-se que o bit k_{15} da chave é 0 e então dividi-se todos os traços em dois grupos. O primeiro grupo corresponde aos traços em que não há variação de

estado lógico no primeiro bit do barramento de dados no deslocamento do ciclo 526 para o ciclo 527 durante o algoritmo criptográfico. Já o segundo grupo corresponde aos traços em que há variação. Uma vez definidos os dois grupos, calcula-se a média de todos os traços de cada grupo, obtendo dois traços médios. Subtraindo-se um traço médio pelo outro obtêm-se um traço diferencial. Em seguida, repete-se o processo, mas supondo que o bit k_{15} da chave é 1. Novamente, dividi-se todos os traços em dois grupos, gera-se dois traços médios e finalmente um novo traço diferencial. Dessa forma tem-se dois traços diferenciais, sendo um o resultado da suposição que o bit k_{15} da chave é 0 e outro de que o bit k_{15} da chave é 1. Os dois traços diferenciais tendem a ter seus valores próximos do valor zero. No entanto, o traço diferencial da suposição correta deverá ter um pico muito evidente, pico este que representa a potência consumida no momento em que os dois bits de dados analisados deslocam pelo barramento de dados durante o processo criptográfico.

A Figura 9 mostra os dois traços diferenciais obtidos por DPA para a suposição de que $k_{15} = 0$ e $k_{15} = 1$. É possível perceber que a suposição de que $k_{15} = 1$ gerou um pico evidente, indicando que essa é a suposição correta.

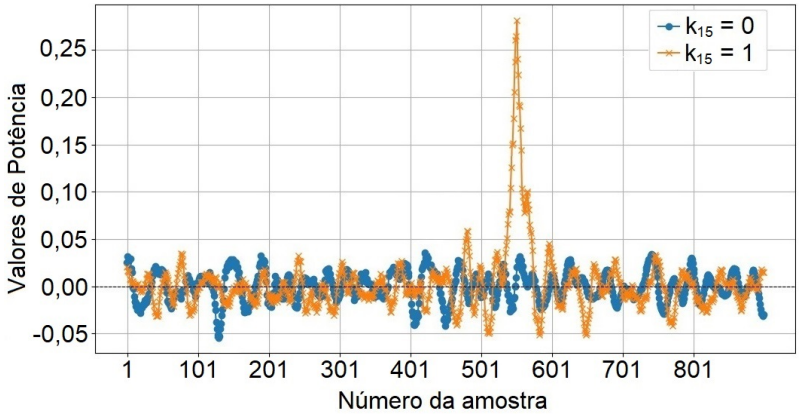

Figura 9. Diferença dos traços médios.

Uma vez que o bit k_{15} da chave é conhecido, a DPA pode ser repetido para descobrir o bit k_{14} da chave, pois o conhecimento do bit k_{15} da chave permite conhecer o estado do barramento de dados no ciclo 527, conforme discutido na Seção 4. Os traços podem ser novamente divididos em dois grupos baseados na suposição do valor do bit k_{14} da chave gerando-se quatro traços médios e, por fim, dois traços diferenciais. Estes irão revelar o correto valor do bit k_{14} da chave. Esse processo é então repetido para os demais bits da chave até que se obtenha a chave completa. Devido à este processo ser puramente computacional e não exigir interface com o HCS201, apenas com os dados já coletados, esta etapa é processada de forma muito rápida, na ordem de minutos.

5.4. Aplicando o ataque

A coleta dos dados e DPA, descritas nas Seções 5.2 e 5.3, foi realizada em dois circuitos integrados HCS201 de dois controles remotos de portão eletrônicos de duas empresas distintas. Após a execução do DPA, foram obtidas as chaves criptográficas supostamente utilizadas em cada um dos controles remotos analisados. A validação do sucesso do ataque foi feita utilizando a chave extraída para decifrar algumas das 10 mil transmissões

criptografadas capturadas durante os acionamentos do HCS201. Para isso, foi implementada a função de decifração do KeeLoq em Python e verificou-se que, após a decifração, os valores do contador presentes nos dados transmitidos seguiam uma sequência coerente, conforme esperado. Além disso, a extração bem-sucedida da chave foi confirmada ao comparar os dez primeiros bits do número serial decifrado com o número serial transmitido em claro, demonstrando que a chave recuperada corresponde exatamente à utilizada pelo chip para cifrar os dados.

Após a confirmação da chave extraída, foram analisados dados de outros controles remotos das mesmas empresas utilizando um receptor de radiofrequência, com objetivo de verificar se a mesma chave criptográfica era empregada em diferentes dispositivos de um mesmo fabricante. Os resultados mostraram que ambas as empresas analisadas utilizam a mesma chave criptográfica em todos os seus controles remotos, sem a aplicação de qualquer método de derivação de chave para proteção adicional do sistema criptográfico. Essa prática representa uma vulnerabilidade crítica, pois a extração bem-sucedida da chave de um único controle remoto compromete todos os dispositivos da mesma marca, permitindo ataques de clonagem e pirataria em larga escala. Dessa forma, qualquer adversário capaz de recuperar a chave de um controle específico pode replicá-la para outros dispositivos do mesmo fabricante, comprometendo toda a segurança do sistema.

Uma vez que se conheça a chave criptográfica, um atacante é capaz de capturar o sinal de um transmissor e, a partir dele, calcular os próximos sinais que o transmissor irá transmitir quando for novamente acionado. Dessa forma, pode-se acessar o receptor transmitindo o mesmo sinal que o transmissor irá transmitir futuramente quando for novamente acionado e até mesmo retirar o acesso do transmissor cujo sinal foi capturado bastando para isso calcular e transmitir o valor do contador com 128 incrementos, conforme discutido na Seção 3.2.1, apagando o transmissor de sua lista de acesso.

5.5. Mitigação

Para mitigar os efeitos do ataque de DPA apresentado neste trabalho em um sistema de segurança baseado no KeeLoq, deve-se utilizar um processo de derivação de chaves baseado, por exemplo, em função de derivação de chaves baseada em HMAC, do inglês HMAC-based key derivation function (HKDF). Este processo consiste em gerar uma chave única K_D, chamada de chave do dispositivo, para cada transmissor (controle remoto) a partir da chave mestra K_M e do serial de cada transmissor. A chave K_M ainda deve ser gravada no receptor, permitindo assim o acionamento do portão eletrônico pelo transmissor. Note que é necessário que o receptor tenha conhecimento do HKDF usado para que o mesmo seja capaz de obter a chave K_D com base no serial recebido no pacote. Neste cenário, partindo do princípio que o HKDF é uma *one-way function* segura e que K_M seja apenas conhecida pela empresa, uma extração de K_D não revela nada sobre K_M. Portanto, um atacante que aplica um ataque DPA bem sucedido em um controle remoto apenas será capaz de acionar um receptor, aquele que é seu par, mas não será capaz de atacar outros pares de transmissor/receptor, exceto no caso em que também faça a quebra deste outro transmissor. É importante notar que essa mitigação não se aplica aos dispositivos já comercializados, tendo em vista que é inviável realizar a regravação de todos eles. Sendo assim, os sistemas de segurança que utilizam os chips HCS201 sem qualquer derivação de chaves ficarão expostos mesmo que apenas um único transmissor tenha sua chave extraída, tendo em vista que, neste caso, esta chave funciona como uma chave

mestra.

Uma outra forma de mitigação possível é reprojetar o HCS201 para que o mesmo não seja vulnerável à ataques por DPA. Para isso, o hardware deve ser remodelado para que seu consumo de potência seja constante ou que seu consumo não revele nada sobre a chave criptográfica, por exemplo, usando mascaramento. No entanto, assim como a mitigação anterior, esta mitigação necessitaria da troca de todos os equipamentos já comercializados, não sendo uma solução viável. Porém, tal medida poderia proteger futuros sistemas de segurança que ainda serão comercializados. Observa-se então que neste caso, a combinação das mitigações propostas é um caminho conservador para garantir a segurança de sistemas baseados no KeeLoq.

6. Conclusão

Este trabalho demonstrou um ataque de baixo custo ao circuito integrado HCS201, utilizado em sistemas de controle de acesso baseados no KeeLoq. Através de DPA, foi possível extrair a chave criptográfica do dispositivo, confirmando a vulnerabilidade da implementação. Diferentemente de abordagens anteriores, que dependem de equipamentos de alto custo, a metodologia proposta emprega hardware acessível, viabilizando a realização do ataque com recursos limitados. Os experimentos ainda mostraram que os fabricantes analisados utilizam a mesma chave criptográfica em todos os seus dispositivos, tornando-os suscetíveis a clonagem e pirataria em larga escala. Como mitigação, foi sugerida a derivação de chaves baseada em HKDF, garantindo chaves únicas para cada transmissor e impedindo que a extração de uma única chave comprometa todo o sistema. Os resultados ressaltam a importância de considerar ataques por canal lateral no desenvolvimento de dispositivos embarcados. Como trabalhos futuros, pretende-se avaliar contramedidas de software que possam mitigar esse tipo de ataque sem necessidade de modificações no hardware.

Referências

[Bogdanov 2007] Bogdanov, A. (2007). Attacks on the keeloq block cipher and authentication systems. In *3rd Conference on RFID Security*, volume 2007.

[Courtois et al. 2008] Courtois, N. T., Bard, G. V., and Wagner, D. (2008). Algebraic and slide attacks on keeloq. In *Fast Software Encryption: 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers 15*, pages 97–115. Springer.

[Eisenbarth et al. 2008] Eisenbarth, T., Kasper, T., Moradi, A., Paar, C., Salmasizadeh, M., and Shalmani, M. T. M. (2008). On the power of power analysis in the real world: A complete break of the keeloq code hopping scheme. In *Advances in Cryptology–CRYPTO 2008: 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings 28*, pages 203–220. Springer.

[Indesteege et al. 2008] Indesteege, S., Keller, N., Dunkelman, O., Biham, E., and Preneel, B. (2008). A practical attack on keeloq. In *Advances in Cryptology–EUROCRYPT 2008: 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings 27*, pages 1–18. Springer.

[Ji et al. 2023] Ji, Y., Wang, R., Ngo, K., Dubrova, E., and Backlund, L. (2023). A side-channel attack on a hardware implementation of crystals-kyber. In *2023 IEEE European Test Symposium (ETS)*, pages 1–5. IEEE.

[Kocher et al. 1999] Kocher, P., Jaffe, J., and Jun, B. (1999). Differential power analysis. In *Advances in Cryptology—CRYPTO'99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, August 15–19, 1999 Proceedings 19*, pages 388–397. Springer.

[Kocher 1996] Kocher, P. C. (1996). Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Advances in Cryptology—CRYPTO'96: 16th Annual International Cryptology Conference Santa Barbara, California, USA August 18–22, 1996 Proceedings 16*, pages 104–113. Springer.

[Marneweck 1996] Marneweck, K. (1996). *An introduction to keeloq code hopping*. Microchip Technology Inc. Rev. 1.0.

[Microchip 2001] Microchip, T. I. (2001). *HCS201 - KEELOQ® Code Hopping Encoder*. Microchip Technology Inc. Rev. 1.0.

[Paar et al. 2009] Paar, C., Eisenbarth, T., Kasper, M., Kasper, T., and Moradi, A. (2009). Keeloq and side-channel analysis-evolution of an attack. In *2009 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pages 65–69. IEEE.

[Soto-Cruz et al. 2024] Soto-Cruz, J., Ruiz-Ibarra, E., Vázquez-Castillo, J., Espinoza-Ruiz, A., Castillo-Atoche, A., and Mass-Sanchez, J. (2024). A survey of efficient lightweight cryptography for power-constrained microcontrollers. *Technologies*, 13(1):3.

[Sun et al. 2025] Sun, P., Wan, Y., Wu, Z., Fang, Z., and Li, Q. (2025). A survey on privacy and security issues in iot-based environments: Technologies, protection measures and future directions. *Computers & Security*, 148:104097.

[Thabit et al. 2023] Thabit, F., Can, O., Aljahdali, A. O., Al-Gaphari, G. H., and Alkhzaimi, H. A. (2023). Cryptography algorithms for enhancing iot security. *Internet of Things*, 22:100759.