UNIVERSIDADE ESTADUAL PAULISTA “JÚLIO DE MESQUITA FILHO”
FACULDADE DE ENGENHARIA DE SÃO JOÃO DA BOA VISTA
BACHARELADO EM ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES

MARIANA TEIXEIRA CARDOSO

UTILIZAÇÃO DO GALOIS COUNTER MODE PARA PROVIMENTO DE AUTENTICAÇÃO E SINCRONIZAÇÃO POR PILOTOS EM ESTRATÉGIAS DE CRIPTOGRAFIA DE SINAIS

SÃO JOÃO DA BOA VISTA
2022

MARIANA TEIXEIRA CARDOSO

UTILIZAÇÃO DO GALOIS COUNTER MODE PARA PROVIMENTO DE AUTENTICAÇÃO E SINCRONIZAÇÃO POR PILOTOS EM ESTRATÉGIAS DE CRIPTOGRAFIA DE SINAIS

Trabalho de Conclusão de Curso apresentado à Universidade Estadual Paulista "Júlio de Mesquita Filho" como requisito para obtenção de título de Bacharel em Engenharia Eletrônica e de Telecomunicações

Orientador: Prof. Dr. Marcelo Luís Francisco Abbade
Coorientadora: Profa. Dra. Cintya Wink de Oliveira Benedito

SÃO JOÃO DA BOA VISTA
2022

C268u

Cardoso, Mariana Teixeira

Utilização do Galois Counter Mode para provimento de autenticação e sincronização por pilotos em estratégias de criptografia de sinais / Mariana Teixeira Cardoso. -- São João da Boa Vista, 2022

49 p.

Trabalho de conclusão de curso (Bacharelado - Engenharia de Telecomunicações) - Universidade Estadual Paulista (Unesp), Faculdade de Engenharia, São João da Boa Vista

Orientador: Marcelo Luís Francisco Abbade

Coorientadora: Cintya Wink de Oliveira Benedito

1. Criptografia. 2. Segurança de sistemas. 3. Processamento de sinais Técnicas digitais. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Engenharia, São João da Boa Vista. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

UNIVERSIDADE ESTADUAL PAULISTA “JÚLIO DE MESQUITA FILHO”
FACULDADE DE ENGENHARIA - CÂMPUS DE SÃO JOÃO DA BOA VISTA
GRADUAÇÃO EM ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES

TRABALHO DE CONCLUSÃO DE CURSO

UTILIZAÇÃO DO GALOIS COUNTER MODE PARA PROVIMENTO DE AUTENTICAÇÃO E SINCRONIZAÇÃO POR PILOTOS EM ESTRATÉGIAS DE CRIPTOGRAFIA DE SINAIS

Aluno: Mariana Teixeira Cardoso
Orientador: Prof. Dr. Marcelo Luís Francisco Abbade

Banca Examinadora:
- Marcelo Luís Francisco Abbade (Orientador)
- Edgar Eduardo Benitez Olivo (Examinador)
- Wilian Miranda dos Santos (Examinador)

A ata da defesa com as respectivas assinaturas dos membros encontra-se no prontuário do aluno (Expediente nº 081/2022)

São João da Boa Vista, 13 de dezembro de 2022

AGRADECIMENTOS

Ao CNPQ por financiar o projeto de iniciação científica que antecedeu este trabalho, por meio da Bolsa PIBIC (proposta 5101).

À professora Cintya, que me introduziu ao mundo da criptografia e me guiou com muito carinho e atenção durante todos os anos da graduação, além de ser uma mulher maravilhosa e exemplar em tudo que faz. Ao professor Marcelo, que se desdobrou para me fazer entender sobre criptografia de sinais e sempre acreditou no meu potencial, me incentivando mesmo quando eu tinha certeza que nada mais daria certo. Aos dois, minha gratidão eterna!

Aos professores que me ajudaram na caminhada, corrigindo e ensinando. Gostaria de agradecer especialmente ao professor Ivan por todos os ensinamentos do último ano.

Agradecimento também ao Raphael Parreira e à Stephanie Lima, meus eternos "barões" e parceiros de graduação. Ao Raphael, obrigada por me ensinar eletrônica, por passar horas dentro do laboratório para que eu entendesse um circuito simples e, depois disso, fortalecer nossa amizade. À Stephanie, por fazer a vida mais leve e que vale a pena ser vivida, por me apoiar e estar presente em todos os momentos difíceis e também nos momentos felizes. Sem vocês, nada seria possível.

Ao amigo Marcelo Nogueira que me ajudou tanto durante a realização deste trabalho, me mostrando códigos, tirando dúvidas e dando opiniões. Também a todos os amigos que fiz durante a graduação, por me ajudarem e me incentivarem.

Agradeço aos meus familiares, que entenderam e me apoiaram sempre. Aos meus avós, que são as pessoas que mais acreditam no meu sucesso e vibram com minhas conquistas, sem medir esforços para me ajudar. À minha mãe, por aguentar tudo, ser compreensiva, parceira, por sofrer e se alegrar comigo. Ao meu namorado Arthur, que sempre me ajuda e está comigo, acreditando e incentivando.

Gostaria de agradecer à Deus, que nunca me abandonou e esteve presente quando ninguém mais estava. Sua presença sempre me deu forças para continuar.

Por fim e não menos importante, ao Boo, meu cachorro, irmão e parceiro. Obrigada por ser meu ponto de paz.

RESUMO

A segurança dos dados enviados e recebidos é tópico atual e amplamente discutido, já que a cada dia são mais utilizados os meios tecnológicos para transmissão de dados confidenciais. O processo de segurança é baseado nos princípios de criptografia, que tem como propósito promover a confidencialidade, garantindo que somente remetente e destinatário da mensagem possam entendê-la. As técnicas de criptografia são aplicadas em diversas situações e protocolos e estão presentes em todas as camadas da pilha de protocolos TCP/IP. Esse trabalho utiliza técnicas de criptografia de dados para realizar autenticação de um sinal obtido por técnicas de criptografia de sinais. Na criptografia de dados é aplicado um algoritmo de chave simétrica, o *Advanced Encryption Standard* (AES), com um modo de operação especificado, o *Galois Counter Mode* (GCM), para gerar uma *tag* que somente transmissor e receptor conseguem calcular. Já na criptografia de sinais foram apresentadas a criptografia de sinais modulados e de sinais em banda-base, além da criptografia quântica. Especificamente, uma versão simplificada da criptografia de sinais em banda-base denominada criptografia de codificação espectral de fase e embaralhamento intracanal por processamento digital de sinais (S*pectral Phase Encoding and Scrambling Cryptography by Digital Signal Processing,* DSP-SPE-Scr) foi utilizada. Por meio dela, foram gerados os sinais criptografados que foram concatenados com as *tags* obtidas na criptografia de dados. Com o uso das *tags* também foi possível garantir sincronização do momento de amostragem entre receptor e transmissor. Os resultados indicaram que a autenticação por meio da técnica descrita é possível mesmo considerando ruídos no canal. Para o sinal criptografado, foram obtidos valores de taxa de erro de bit (*Bit Error Rate,* BER) que variaram menos de uma ordem de grandeza para diferentes tamanhos de *tag*. Sendo assim, do ponto de vista dos requisitos de camada física, *tags* com apenas 32 bits de comprimento (o menor tamanho analisado) podem ser utilizadas. No entanto, do ponto de vista de segurança, pode ser conveniente usar tamanhos maiores de *tag*.

Palavras-chave: Criptografia, Segurança, Processamento digital de sinais

ABSTRACT

The security of data sent and received is a current and widely discussed topic, since technological means are being used every day for the transmission of sensitive data. The security process is based on the principles of cryptography, which aims to promote confidentiality, ensuring that only the sender and receiver of the message can understand it. Cryptography techniques are applied in different situations and protocols and are present in all layers of the TCP/IP protocol stack. The present study uses data encryption techniques to authenticate a signal obtained by signal encryption techniques. Data encryption is applied using a symmetric key algorithm, Advanced Encryption Standard (AES), with a specified mode of operation, Galois Counter Mode (GCM), to generate a tag that only the transmitter and receiver can calculate. For signal encryption, the encryption of modulated signals and baseband signals were introduced, in addition to quantum cryptography. Specifically, a version of baseband signal encryption called *spectral phase encoding and scrambling cryptography by digital signal processing* (DSP-SPE-Scr) was used. Through it, encrypted signals were generated and concatenated with the tags generated in data encryption. Using tags, it was also possible to guarantee synchronization between receiver and transmitter. The obtained results indicated that authentication using the described technique is possible even considering noisy channels. For the encrypted signal, Bit Error Rate (BER) values that varied by less than an order of magnitude were obtained for different tag sizes. Therefore, from the point of view of physical layer requirements, tags that are only 32 bits long (the smallest size analyzed) can be used. However, from a security point of view, it may be convenient to use larger tag sizes.

Keywords: Cryptography, Security, Digital signal processing

LISTA DE FIGURAS

LISTA DE SIGLAS E ABREVIATURAS

AAD	Additional Authenticated Data (Dados Autenticados Adicionais)
AES	Advanced Encryption Standard (Padrão de Criptografia Avançada)
AWGN	Additive White Gaussian Noise (Ruído Aditivo Gaussiano Branco)
BER	Bit Error Rate (Taxa de Erro de Bit)
CTR	Counter Mode (Modo Contador)
dB	Decibel
DNS	Domain Name System (Sistema de Nomes de Domínio)
DSP	Digital Signal Processing (Processamento Digital de Sinais)
DSP-SPE-Scr	Spectral Phase Encoding and Scrambling Cryptography by Digital Signal Processing (Criptografia de Codificação Espectral de Fase e Embaralhamento Intracanal por Processamento Digital de Sinais)
FFT	Fast Fourier Transform (Transformada Rápida de Fourier)
GCM	Galois Counter Mode (Modo Contador de Galois)
GHASH	Função do GCM
GMAC	Versão do GCM para autenticação
HTTP	Hypertext Transfer Protocol (Protocolo de Transferência de Hipertexto)
I	Componente em fase do sinal
IV	Initialization Vector (Vetor de Inicialização)
IFFT	Inverse Fast Fourier Transform (Transformada Inversa Rápida de Fourier)
IF	Invocation Field (Campo de Invocação)
IP	Internet Protocol (Protocolo de Internet)
IPSec	Internet Protocol Security (Segurança do Protocolo de Internet)
ISO	International Organization for Standardization (Organização Internacional de Normalização)
MAC	Message Authentication Code (Autenticador de Mensagem)
OSI	Open System Interconnection (Interconexão de Sistemas Abertos)
PGP	Pretty Good Privacy (Privacidade Muito Boa)
Q	Componente em quadratura do sinal
QKD	Quantum Key Distribution (Distribuição de Chaves Quânticas)
QPSK	Quadrature Phase Shift Keying (Modulação por Deslocamento de Fase em Quadratura)

RSA	Rivest-Shamir-Adleman
SLM	Spatial Light Modulator (Modulador Espacial de Luz)
SMTP	Simple Mail Transfer Protocol (Protocolo de Transferência de Correio Simples)
SNR	Signal-to-Noise Ratio (Razão Sinal-Ruído)
SSL	Secure Sockets Layer (Camada de Soquetes Seguros)
TCP	Transmission Control Protocol (Protocolo de Controle de Transmissão)
TSL	Transport Layer Security (Segurança da Camada de Transporte)
TON	Transparent Optical Network (Rede Óptica Transparente)
UDP	User Datagram Protocol (Protocolo de Datagrama do Usuário)
XOR	Ou-exclusivo

SUMÁRIO

1 – INTRODUÇÃO

1.1 Modelo OSI, conjunto de protocolos TCP/IP e técnicas de Criptografia

A padronização de protocolos é a responsável por permitir a comunicação entre dois sistemas diferentes, como por exemplo dispositivos fornecidos por diferentes fabricantes. Em 1984, a Organização Internacional de Normalização (*International Organization for Standardization*, ISO) estabeleceu uma estrutura em camadas para realizar a padronização, denominada modelo de referência para Interconexão de Sistemas Abertos (*Open System for Interconnections*, OSI). Cada uma das 7 camadas do modelo é responsável por um conjunto de funções, de modo que a ação de enviar uma mensagem de um dispositivo A para um dispositivo B pode envolver diversas camadas. A relação entre as camadas é bidirecional, por exemplo, se um dispositivo é o receptor, sua camada de rede (camada 3) recebe informações da camada de enlace de dados (camada 2) e envia informações para a camada de transporte (camada 4). Já durante a transmissão, a camada de rede recebe informações da camada de transporte e envia para a camada de enlace de dados. Para a comunicação entre diferentes dispositivos, cada camada se comunica com a camada correspondente, como por exemplo a camada de rede do dispositivo A se comunica com a camada de rede do dispositivo B (FOROUZAN, 2008) e essa comunicação é realizada por meio dos protocolos. É importante notar que o modelo OSI é responsável por servir de referência e definir as funções de cada camada, mas a implementação de fato dessas funções é realizada pelos protocolos, que podem variar de acordo com diferentes fabricantes. Um exemplo amplamente utilizado na internet é o conjunto de protocolos TCP/IP (*Transmission Control Protocol/Internet Protocol)*, que condensa algumas camadas do modelo OSI.

Cada camada do conjunto TCP/IP utiliza seus próprios protocolos: a camada de aplicação é responsável por fornecer a interface com o usuário e os serviços por ele utilizados e inclui protocolos como o DNS (*Domain Name System*), SMTP (*Simple Mail Transfer Protocol*) e HTTP (*Hypertext Transfer Protocol*). A camada de transporte faz o envio dos dados por meio dos protocolos TCP e UDP (*User Datagram Protocol*) e a camada de internet determina a melhor rota para que os dados sejam enviados, utilizando por exemplo o protocolo IP. Já na camada de enlace de dados é feito o controle de hardware e, no último momento, os dados são transmitidos para o dispositivo com o qual

é desejado se comunicar. As camadas superiores contam com protocolos específicos para adicionar segurança: a camada de aplicação possui o PGP (*Pretty Good Privacy*) para adicionar segurança ao SMTP, a camada de transporte conta com o SSL (*Secure Sockets Layer)* e o TLS (*Transport Layer Security*) e a camada de internet utiliza o IPSec (*Internet Protocol Security*).

A Figura 1 indica um modelo em camadas modificado em relação ao conjunto de protocolos TCP/IP, pois adiciona uma camada física. Além disso, são indicados alguns protocolos de segurança que podem ser utilizados. Como cada camada utiliza informações da camada imediatamente anterior, então as medidas para garantir segurança em uma camada são enviadas para as outras. É possível observar também que os protocolos das camadas superiores utilizam técnicas de criptografia de dados, enquanto na camada física os dados são convertidos para sinais e, comercialmente, não são empregadas técnicas para garantir segurança. Assim, os sinais são enviados de um meio ao outro sem criptografia.

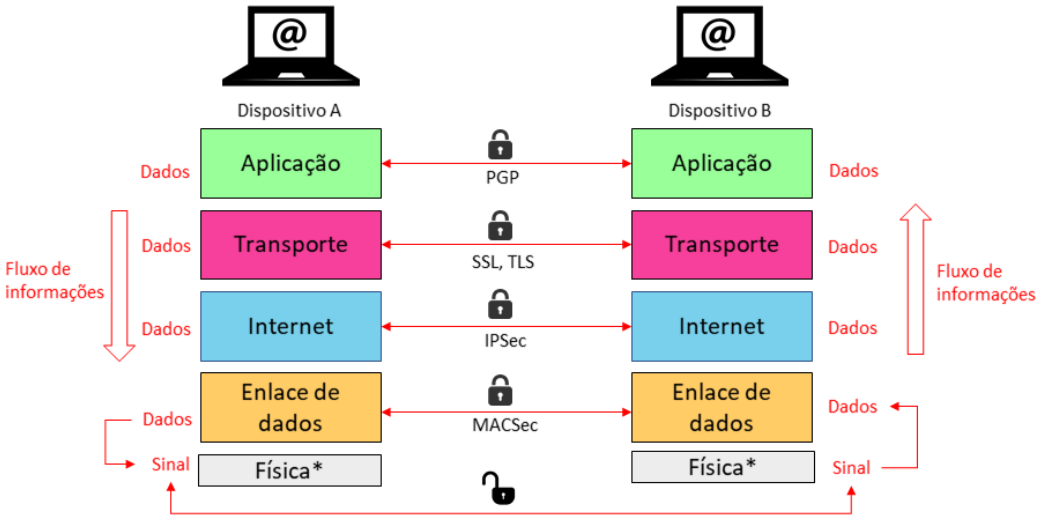

Figura 1 – Modelo em camadas e protocolos de segurança relacionados. Fonte: baseado em (ABBADE *et al.*, 2021).

Foram propostas diversas soluções para criptografar sinais que são enviados por enlaces de telecomunicações ou por redes transparentes, como por exemplo em (CORNEJO, PEREZ e TOCNAYE, 2007), (SANTOS, 2020), (NOGUEIRA, 2022), (YANG *et al.*, 2016) e (ABBADE *et al.*, 2019).

Já a criptografia de dados é empregada para garantir a segurança ao nível de bits e seus algoritmos criptográficos podem ser divididos em cifras assimétricas, também

conhecidas como algoritmos de chave pública, e cifras simétricas, também conhecidas como algoritmos de chave privada. Para as cifras assimétricas, os usuários utilizam uma chave pública para criptografar e uma chave privada para descriptografar. Um algoritmo de cifra assimétrica bastante utilizado foi estabelecido por Rivest-Shamir-Adleman: RSA (RIVEST, SHAMIR e ADLEMAN, 1978). As cifras assimétricas podem ser utilizadas para que usuários criptografem e troquem entre si uma chave privada, a qual pode ser utilizada depois para criptografar as mensagens por meio das cifras simétricas.

As cifras simétricas utilizam uma mesma chave privada para criptografar e descriptogafar dados: o usuário que deseja transmitir a mensagem aplica o algoritmo de encriptação simétrico aos dados, utilizando uma chave *k* que só deve ser conhecida pelo transmissor e pelo receptor da mensagem. Com esse procedimento, é possível obter um texto cifrado que é enviado até o receptor. Por sua vez, o receptor aplica o algoritmo simétrico para desencriptação utilizando a chave *k*, recuperando a mensagem. Os protocolos SSL, TLS e IPSec utilizam cifras simétricas. Uma cifra simétrica muito utilizada é conhecida como *Advanced Encryption Standard* (AES) e será discutida em detalhes posteriormente neste trabalho.

Para a camada física é necessário utilizar algoritmos para fazer criptografia de sinais, que pode ser dividida em criptografia baseada em física quântica e baseada em física clássica. A criptografia baseada em física quântica faz uso de partículas quânticas e geralmente é limitada para aplicações que necessitam pouca quantidade de informações, como por exemplo a transmissão de chaves, chamada distribuição quântica de chaves (*Quantum Key Distribution,* QKD). Já para a criptografia baseada em física clássica, os sinais modulados podem ser criptografados utilizando técnicas que envolvem equipamentos como lentes e moduladores de luz. Com isso, o sinal é dividido em fatias espectrais e a encriptação é feita adicionando uma diferença de fase em cada fatia.

Como uma das desvantagens da criptografia de sinais modulados é utilizar equipamentos relativamente caros, foi proposto fazer a criptografia dos sinais em banda-base, trabalhando com o sinal no domínio digital por meio de processamento digital de sinais. Por exemplo, conforme indicado em (SANTOS, 2020), é adicionada a diferença de fase por meio de técnicas de programação na linguagem Matlab®. Na Figura 2, o diagrama divide os tipos de criptografia citados neste trabalho e seus principais exemplos.

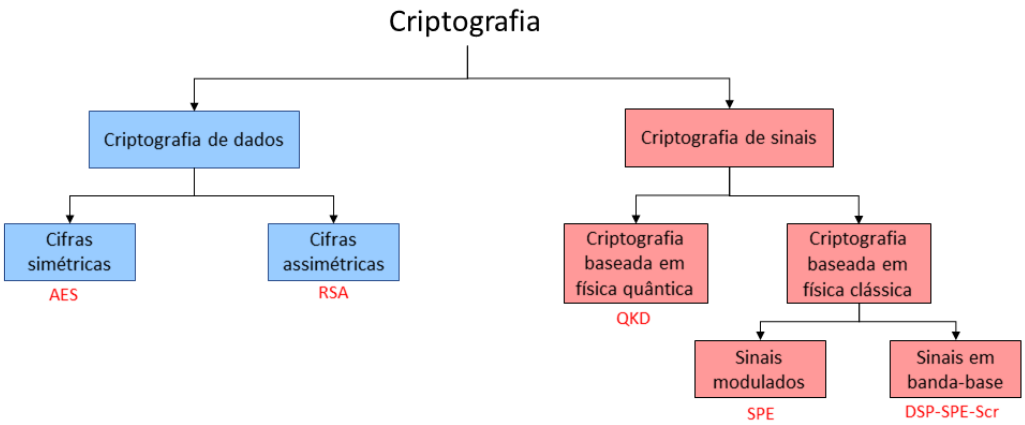

Figura 2 – Diagrama de blocos para os tipos de criptografia.

1.2 Identificação do instante de amostragem, autenticação e objetivos

Para recuperar um sinal criptografado em banda-base que foi enviado por meio de uma rede óptica por exemplo, é necessário identificar qual o exato momento para iniciar sua amostragem. Uma ferramenta que possibilita essa identificação é acrescentar um cabeçalho aos dados: quando o cabeçalho termina, significa que já é o momento de início da amostragem do sinal. No entanto, acrescentar um cabeçalho também significa desperdiçar banda com dados que não possuem informação relevante. Além disso, maiores bandas permitem passagem de mais ruído.

Um bom uso, portanto, seria aproveitar as informações desse cabeçalho para realizar a autenticação do sinal, garantindo que o remetente é de fato quem diz ser e não um indivíduo mal-intencionado. Assim, o objetivo desse trabalho é unir técnicas de criptografia de dados e criptografia de sinais: por meio de um modo de operação da cifra simétrica AES, o *Galois Counter Mode* (GCM), são geradas *tags* de autenticação, que devem ser convertidas em sinais e concatenadas ao sinal criptografado em banda-base. Desse modo, as *tags* de autenticação servem como cabeçalho para identificar o instante de amostragem e também servem como autenticação, já que somente transmissor e receptor são capazes de gerar as *tags* por meio da chave privada *k*. Além disso, o trabalho busca avaliar os tamanhos que as *tags* de autenticação podem assumir, considerando requisitos de segurança e da identificação correta da amostragem.

1.3 Organização do trabalho

O restante desse trabalho é organizado conforme indicado a seguir.

O Capítulo 2 aborda conceitos de criptografia de dados, entrando em detalhes sobre o AES e seu modo de operação utilizado para a autenticação, o GCM. No Capítulo 3 são indicadas técnicas de criptografia de sinais, diferenciando seus tipos e aprofundando um pouco mais sobre a criptografia de sinais em banda-base, que é utilizada neste trabalho. Além disso, o Capítulo 3 contém a proposta de autenticação, mostrando como foi planejada a utilização de *tags* de autenticação. No Capítulo 4 são apresentados os códigos utilizados e os resultados são indicados e comentados. No Capítulo 5 é concluído o trabalho, mostrando seus pontos principais e considerações finais.

2 – CONCEITOS DE CRIPTOGRAFIA DE DADOS E AUTENTICAÇÃO

A criptografia é responsável por promover a confidencialidade de dados, garantindo que somente remetente e destinatário da mensagem consigam entendê-la. Diversos algoritmos de encriptação podem ser utilizados, mas um deles é particularmente útil para este trabalho: o AES.

Este capítulo introduz uma visão geral sobre o AES, seu funcionamento e complexidade. Depois, cita a necessidade dos modos de operação e entra em detalhes sobre o *Counter Mode* (CTR). Na Seção 2.2, é comentado sobre o conceito de autenticação e apresentado o *Galois Counter Mode* (GCM), especificando seu funcionamento, parâmetros necessários, funções e operações utilizadas.

2.1 Advanced Encryption Standard (AES)

O AES é um algoritmo de encriptação amplamente utilizado comercialmente. É uma cifra simétrica de bloco que foi introduzida por dois criptógrafos belgas: Joan Daemen e Vincent Rijmen. (DAEMEN e RIJMEN, 2002) e que opera em blocos com tamanho de 128 bits e chaves variando entre 128, 192 e 256 bits. Por se tratar de uma cifra simétrica, o AES utiliza uma única chave *k* para criptografar e descriptografar os dados e uma de suas operações permite gerar sub-chaves a partir de *k.*

Para fazer o processo de encriptação, cada rodada do AES utiliza uma sub-chave diferente e, em cada rodada, são implementados os seguintes estágios: *AddRoundKey*, *SubBytes, ShiftRows* e *MixColumns.* Para uma chave de 128 bits são necessárias 10 rodadas, para uma chave de 192 bits são 12 rodadas e para a chave de 256 bits são utilizadas 14 rodadas. As operações realizadas estão contidas no Corpo de Galois - GF (2^8), de modo que todas as adições realizadas no AES correspondem a uma operação da função lógica ou-exclusivo (*exclusive-or,* XOR) e a operação de multiplicação deve ser feita na forma *a mod b*, na qual *a* equivale ao resultado da multiplicação dos dois valores em forma de polinômio e *b* representa o polinômio irredutível utilizado pelo AES.

O AES é uma cifra de bloco e, portanto, opera sempre com blocos de tamanho fixo. No entanto, muitas vezes os dados a serem criptografados possuem tamanhos diferentes, sendo maiores ou menores que um bloco para encriptação. Nesse caso, faz-se o uso dos modos de operação (PAAR e PELZL, 2010), que são responsáveis por fazer a

encriptação de dados maiores e em alguns casos promover autenticação, garantindo que a mensagem realmente veio do usuário transmissor.

Uma característica importante dos modos de operação é que devem fornecer aleatorização, ou seja, se a mesma mensagem for criptografada duas vezes utilizando a mesma chave, os textos criptografados devem ser diferentes entre si em cerca de metade dos bits. Uma das características que possibilitam a aleatorização é o uso de um vetor de inicialização (*Initialization Vector*, IV).

Os modos de operação estão definidos em (DWORKIN, 2001) e um modo particularmente útil é conhecido como *Counter Mode* (CTR). Nele, é possível produzir uma série de contadores e utilizá-los da seguinte forma: um IV é concatenado com um contador de modo que o vetor concatenado possua o tamanho da cifra de bloco (128 bits no caso do AES) e esse vetor passa pelo algoritmo de encriptação. Depois, é feita uma operação XOR com o primeiro bloco do texto de entrada que contém 128 bits para gerar o primeiro bloco de texto cifrado. Para o próximo bloco, o contador passa por um incremento, o IV se mantém constante e as operações se repetem, conforme indicado na Figura 3.

O contador deve ser escolhido de maneira a produzir uma sequência que não vai se repetir para uma mesma chave (DWORKIN, 2001). É possível escolher uma função que incrementa uma constante, por exemplo 1, a cada novo contador. É importante ressaltar que o novo vetor formado pelo IV e o contador não necessita ser secreto, visto que será encriptado por uma chave secreta.

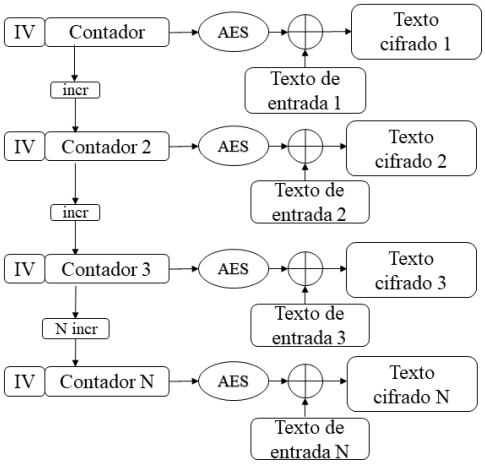

Figura 3 – Diagrama de blocos do CTR.

2.2 Galois Counter Mode (GCM)

O GCM é um modo de operação do AES que utiliza um autenticador de mensagem (*Message Authentication Code*, MAC) para realizar a autenticação da mensagem. O MAC é calculado pelo remetente e anexado à mensagem a ser enviada. O destinatário da mensagem também calcula um MAC através da mensagem que recebeu. Se o MAC calculado pelas duas partes é o mesmo, então pode-se garantir a autenticação, ou seja, a mensagem foi realmente enviada pelo remetente e também garantir a integridade, indicando que não houve alteração do texto cifrado durante a transmissão (PAAR e PELZL, 2010).

Entre as diversas vantagens do GCM estão por exemplo a possibilidade de criptografar em altas velocidades e a autenticação da mensagem. Outro ponto positivo é que o GCM é capaz de autenticar as mensagens mesmo quando não há dados para criptografar. Além disso, se uma *tag* de autenticação MAC é computada para uma mensagem e então uma parte dessa mensagem é modificada, a nova *tag* pode ser obtida com custo computacional proporcional ao número de bits que foram modificados na mensagem, ou seja, os bits iguais não irão requisitar mais nada do sistema (MCGREW e VIEGA, 2004).

O GCM conta com duas operações, encriptação e decriptação, ambas possibilitando o processo de autenticação. Para o processo de encriptação descrito por McGrew e Viega (2004) são utilizadas quatro entradas: a chave (K) cujo tamanho varia de acordo com a cifra de bloco utilizada; um IV que pode ter tamanho variando de 1 até 2^{64} bits; um texto de entrada que pode ter tamanho variando de 0 até $2^{39} - 256$ bits e por fim um vetor de Dados Autenticados Adicionais (*Additional Authenticated Data*, AAD), um bloco que deve ser autenticado mas não criptografado, com tamanho também variando entre 1 e 2^{64} bits. Inserindo as entradas no processo de encriptação, é possível obter o texto cifrado com mesmo tamanho do texto de entrada e também uma *tag* de autenticação, cujo tamanho varia entre 0 e 128 bits.

Para a desencriptação, são necessárias cinco entradas: K, IV, AAD, o texto cifrado e também a *tag* de autenticação. Como resultado, é encontrada somente uma variável, sendo ela ou o texto de entrada ou uma notificação de falha, indicando que as entradas não são autênticas. A saída vai notificar falha sempre que as entradas não forem criadas pela etapa

de encriptação utilizando a mesma chave, ou sempre que os parâmetros de entrada não forem os mesmos utilizados no momento da encriptação.

O IV utilizado é chamado *nonce*, ou seja, um valor numérico que só pode ser utilizado uma vez se for considerada uma mesma chave (PAAR e PELZL, 2010). Os IVs podem ser gerados de forma aleatória, mas deve-se garantir que sejam distintos entre si. Uma das maneiras recomendadas para criar um IV é por meio da construção determinística, na qual um campo fixo e um Campo de Invocação (*Invocation Field*, IF) são concatenados. De acordo com (DWORKIN, 2007), o campo fixo pode ser utilizado para identificar o dispositivo ou o contexto da autenticação e o IF é variado a cada rodada, de modo a identificar cada uma das entradas em um mesmo contexto.

Apesar de os IVs possuírem tamanho arbitrário, é importante notar que para maior segurança e para promover interoperabilidade, eficiência e simplicidade de implementação (DWORKIN, 2007) é sugerido 96 bits de tamanho para o vetor fixo e 32 bits para o vetor de invocação, totalizando 128 bits. Ao utilizar um vetor com tamanho diferente do recomendado, as propriedades únicas não são mantidas e aumentam as chances de ocorrer repetição no valor do vetor concatenado, o que reduz muito a segurança do GCM e, portanto, o inutiliza.

Já o AAD é usado no GCM para proteger informações que devem ser mantidas sem encriptação, como por exemplo endereços IP, portas, números de protocolos e até mesmo outros dados para auxiliar na manipulação do texto de entrada (MCGREW e VIEGA, 2004). Sua utilização é arbitrária, servindo para auxiliar no processo de autenticação.

A confidencialidade e autenticidade são garantidas no GCM e a força da autenticação é determinada pelo tamanho da *tag* de autenticação. As *tags* de 128 bits, por exemplo, são mais seguras que *tags* com 96 bits. Para o caso específico em que o texto de entrada tem tamanho nulo, o GCM age como um MAC no AAD, sendo responsável por fazer somente a autenticação e não a encriptação de dados. Nesse caso, o algoritmo é chamado de GMAC.

O algoritmo para o GCM é idealizado da seguinte forma: os dados são criptografados da mesma maneira que no CTR, seguido do processo para obtenção da *tag* de autenticação. Inicialmente, o campo fixo e o IF são concatenados e passam pela encriptação. Depois, o IF é incrementado, os campos concatenados são criptografados e

o resultado passa por um XOR com o primeiro bloco do texto de entrada. A operação é repetida para os demais blocos, com o IF sendo incrementado novamente a cada bloco. Assim, uma das vantagens do GCM é otimizar tempo, já que conhecendo um IV é possível estabelecer os valores do contador e encriptá-los, restando apenas a operação de XOR com o texto de entrada.

Já a função de *hash* utilizada no GCM, denominada GHASH, é aplicada diversas vezes durante o processo de autenticação e funciona da seguinte forma: o vetor de entrada X, cujo tamanho é dado por $len\ (X) = 128m$ é dividido em m blocos de 128 bits. Cada um dos blocos passa inicialmente por uma operação XOR com um bloco Y_0, composto somente por zeros, resultando no próprio bloco inicial de X. Esse resultado passa por uma multiplicação por uma constante *hash* (H) no Corpo de Galois – GF(2^{128}), com polinômio irredutível $P(x) = x^{128} + x^7 + x^2 + x + 1$. A constante *hash* é obtida por meio da encriptação de um texto de entrada composto por zeros, utilizando a chave *k*. Para a operação de multiplicação, foi criada uma função baseada no algoritmo apresentado por (GUERON e KOUNAVIS, 2010).

Como resultado dessa multiplicação, obtém-se o bloco Y_1. Se $m = 1$, então Y_1 é a saída da função GHASH. Se $m > 1$, então é feito um XOR entre Y_1 e o próximo bloco de X, representado na Figura por X_2. O resultado é novamente multiplicado por *hash*, obtendo Y_2. O procedimento é repetido de forma a serem utilizados todos os m blocos de X e a saída da função GHASH é Y_m, com 128 bits. Na Figura 4, adaptada de (DWORKIN, 2007), a multiplicação pela constante *hash* no GF(2^{128}) é representada por $\cdot H$.

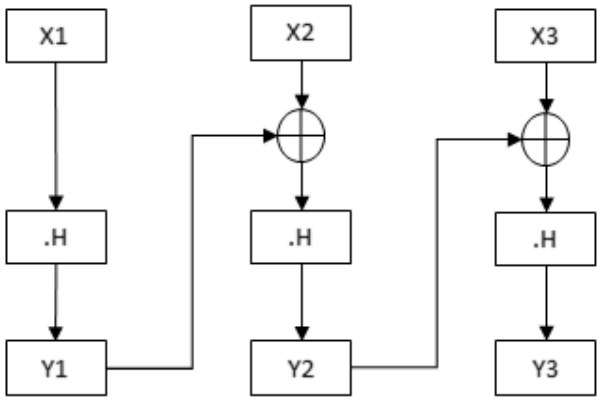

Figura 4 – Função GHASH.

Para o processo de autenticar e, portanto, obter o valor MAC, pode ser utilizado ou não o AAD. Se o bloco estiver presente, deve passar pela função GHASH. A saída de GHASH deve passar por um XOR com o primeiro bloco do texto cifrado, obtido no processo de encriptação. O resultado do XOR deve passar por GHASH e o procedimento deve ser repetido até que o último bloco do texto cifrado tenha passado pelo XOR e pela função GHASH. Essa última saída deve passar por um XOR com um vetor composto pela concatenação de dois outros vetores de 64 bits cada, sendo indicado na Figura 5 como "len(A)||len(C)". O vetor "len(A)" representa o tamanho de AAD, em bits, e o vetor "len(C)" representa o tamanho do texto cifrado (do inglês *ciphertext)*, também em bits. Por exemplo, se o AAD tem tamanho nulo e o texto cifrado tem 128 bits, então o vetor composto representado em hexadecimal vai ser dado conforme a Figura 5. Note que o valor "80" em hexadecimal corresponde a 128 em decimal, indicando o tamanho em bits do texto cifrado.

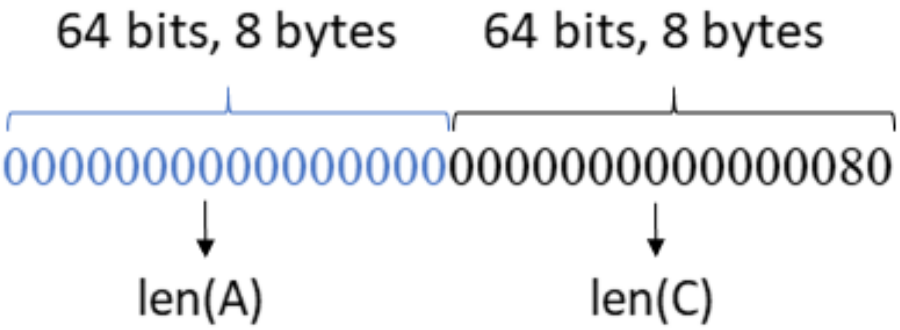

Figura 5 – Exemplo de vetor concatenado.

Novamente, o resultado passa por GHASH. Por fim, a *tag* de autenticação é obtida de um XOR entre o contador inicial encriptado (Ek_0) e o último parâmetro de GHASH obtido. Um diagrama do funcionamento do GCM foi adaptado de (MCGREW e VIEGA, 2004) e está demonstrado na Figura 6.

Para o caso em que não é utilizado texto de entrada e deseja-se somente fazer o processo de autenticação, o processo se resume ao apresentado na Figura 7. Nesse caso, o GCM passa a ser nomeado GMAC.

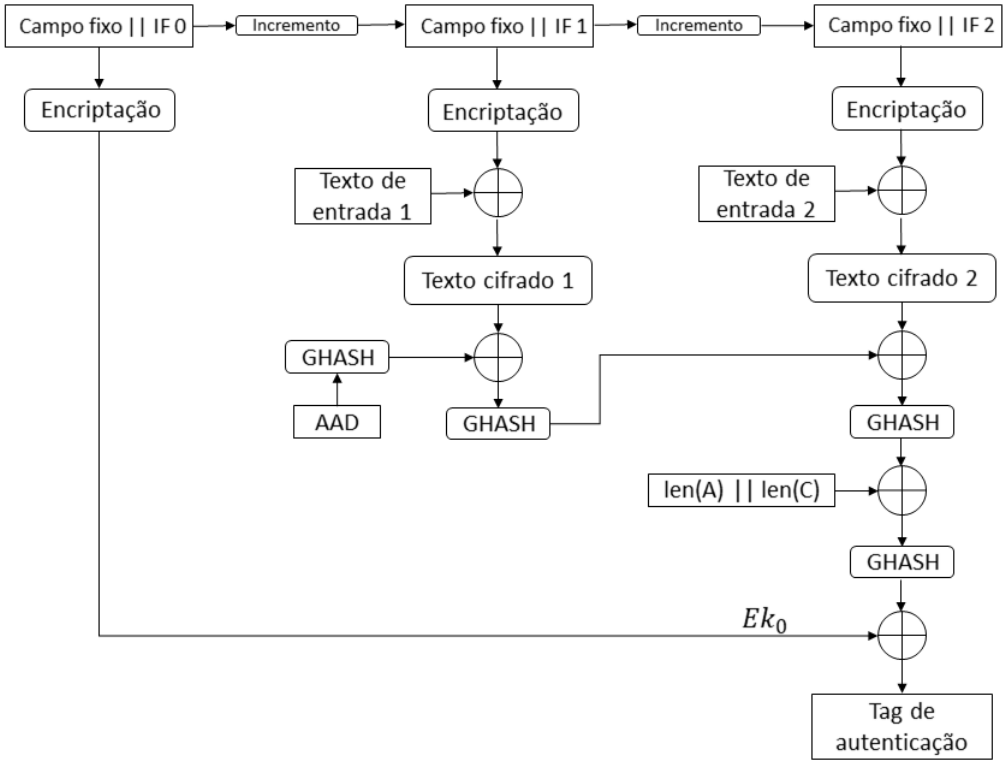

Figura 6 – GCM: diagrama de blocos.

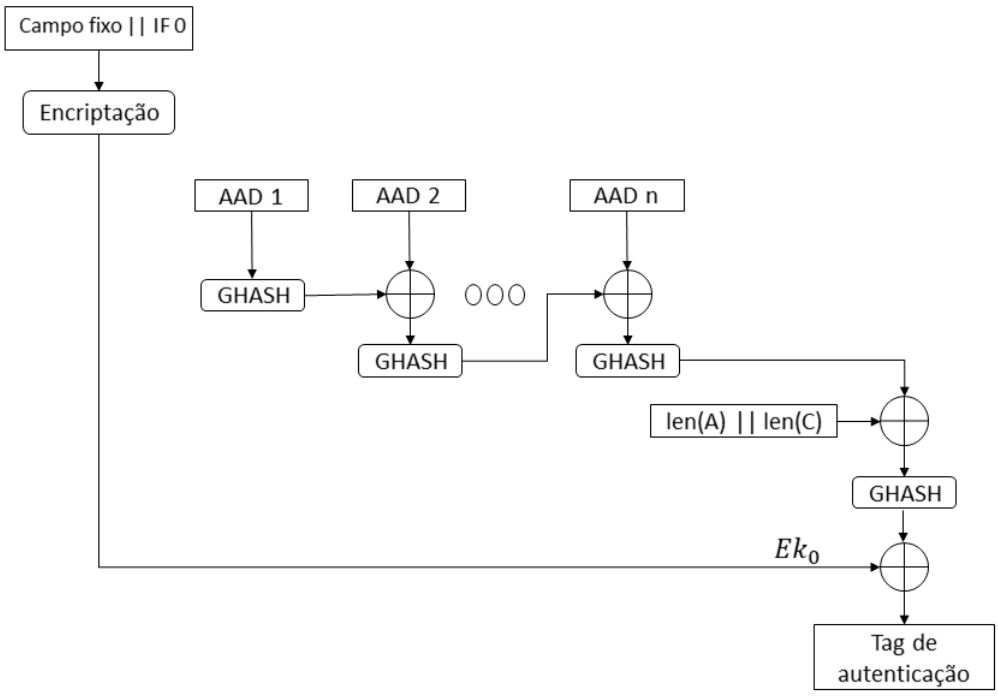

Figura 7 – GMAC: diagrama de blocos.

As operações no GF (2^{128}) podem ser melhor compreendidas e foram implementadas baseando-se no documento fornecido por (GUERON e KOUNAVIS, 2010). A adição, assim como no AES, é obtida por meio de XOR entre os elementos somados. Já a multiplicação apresenta diferenças, visto que seu polinômio irredutível é diferente.

A multiplicação utilizada é chamada *carry-less*, a qual será mostrada a seguir através de um exemplo com polinômios menores que os utilizados no GCM. Considerando que a = 10100010 e b = 10010110, a multiplicação *carry-less* entre os termos ocorre da seguinte forma:

Para cada bit 1 de *a*, o *b* vai passar por um deslocamento para a esquerda correspondente à posição do bit 1. Cada ocorrência do bit 1 em *a* gera um deslocamento correspondente e esses deslocamentos devem passar por um XOR entre si. No exemplo da Figura 8, se for considerado o último bit da esquerda para a direita como posição zero, notam-se bits 1 nas posições 1,5 e 7 de *a*. Os elementos a passarem pelo XOR, portanto, equivalerão ao elemento *b* (em vermelho) com 1 zero adicionado à sua direita, elemento *b* com 5 zeros adicionados à sua direita e elemento *b* com 7 zeros adicionados à sua direita. Assim, ao realizar o XOR entre esses elementos, obtêm-se o resultado da multiplicação *carry-less* entre *a* e *b*. Então, *c* = 101100011101100.

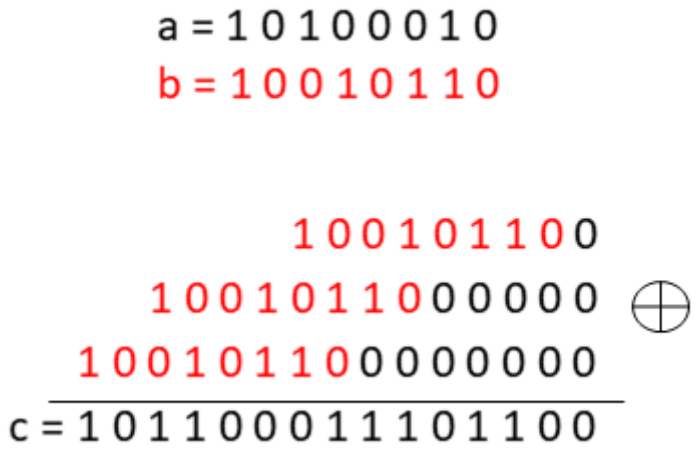

Figura 8 – Multiplicação *carry-less*.

Portanto, é feita a multiplicação *carry-less* entre os elementos de entrada da função GHASH e obtido um vetor de 256 bits conforme o Algoritmo 1 apresentado em (GUERON e KOUNAVIS, 2010) e que foi adaptado para Matlab®. O vetor de 256 bits

deve passar pelo processo de redução, que consiste em fazer uma operação módulo: $a \bmod b$, na qual a é o vetor de 256 bits e b é o polinômio irredutível do GCM, ou seja, deve-se fazer divisão do vetor de 256 bits pelo polinômio e considerar o resto dessa divisão. Um método simplificado para fazer essa operação com polinômios foi descrito pelo Algoritmo 4 em (GUERON e KOUNAVIS, 2010) e foi adotado neste trabalho. Assim, aplicando o algoritmo de redução é obtido o vetor de 128 bits que é o resultado da multiplicação em GF(2^{128}).

Deve-se levar em consideração também uma peculiaridade apresentada pelo GCM: os bits que entram na multiplicação devem ser refletidos, de modo que o resultado na multiplicação em GF(2^{128}) não é o mesmo da multiplicação em GF(2^{128}) para o GCM. Como exemplo: 11100001 refletido para o GCM fica 10000111. Para a saída, os bits da multiplicação também são refletidos e esse procedimento foi utilizado e adaptado no algoritmo desenvolvido.

3 – CRIPTOGRAFIA DE SINAIS

No capítulo anterior foram descritas técnicas de encriptação e autenticação de dados que são aplicadas em diversos protocolos comerciais. Neste capítulo será abordado o modelo OSI e seus protocolos de segurança, justificando a necessidade de criptografar sinais, os quais são enviados através da camada física. Além disso, é apresentada a criptografia quântica, seus benefícios, limitações e usos comuns.

Em seguida, é introduzida a criptografia de sinais baseada em física clássica, que pode ser aplicada em sinais modulados ou sinais em banda-base. Para os sinais modulados são indicadas duas técnicas e apresentadas algumas características. Já para os sinais em banda-base é mostrada a técnica de criptografia de codificação espectral de fase e embaralhamento intracanal por processamento digital de sinais (*Spectral Phase Encoding and Scrambling Cryptography by Digital Signal Processing*, DSP-SPE-Scr), suas vantagens em relação à criptografia de sinais modulados e sua simplificação a ser utilizada neste trabalho.

Por fim, é descrita a proposta de autenticação considerando sinais criptografados em banda-base por meio da simplificação da DSP-SPE-Scr e um cabeçalho que passa pelo GMAC para gerar a *tag* de autenticação.

3.1 Conceitos

O modelo OSI é estruturado em camadas e foi criado com a intenção de padronizar protocolos e, assim, permitir a comunicação entre diferentes sistemas (FOROUZAN, 2008). Diversas técnicas de segurança são empregadas em camadas superiores, mas os dados convertidos em sinais por meio da camada física não são criptografados.

Sinais enviados de um meio a outro sem criptografia podem ser interceptados por indivíduos mal-intencionados: uma fibra óptica pode sofrer ataques intrusivos e não intrusivos, por exemplo ao dobrar a fibra e utilizar um acoplador passivo de grampo (*passive fiber clip-on coupler)* (UEMATSU *et al.*, 2017) ou até mesmo ao usar divisores ópticos (*optical splitters)* para obter informações. Portanto, criptografar os sinais transmitidos é uma maneira de garantir a segurança dessas informações.

A criptografia de sinais pode ser dividida em dois tipos: criptografia baseada em física quântica e baseada em física criptografia clássica. A criptografia baseada em física quântica é conhecida por utilizar partículas quânticas como fótons e ser capaz de detectar

a presença de intrusos, já que é impossível copiar um estado quântico sem alterá-lo (ABBADE *et al.*, 2021). A desvantagem da criptografia quântica é a taxa de transmissão muito baixa, com máximo de 100 kbps para uma distância de 100 km entre as estações utilizadas para a comunicação (LUCAMARINI *et al.*, 2018). Ainda segundo a mesma referência, com o aumento da distância se reduz ainda mais a taxa de transmissão, chegando somente 1 bit por minuto para cerca de 600 km. Portanto, é usual que poucos bits sejam transmitidos utilizando a criptografia quântica e um caso conhecido é o envio de chaves por meio da distribuição de chaves quânticas (*Quantum Key Distribution*, QKD), que tem como representantes mais conhecidos o BB84 (BENNET e BRASSARD, 1984) e o E91 (EKERT, 1991). Uma solução para uso global de QKD inclui satélites e possibilita taxa de transmissão da chave na ordem de kbits/s para distâncias de até 1200 km, considerando a transmissão do satélite até a Terra. Assim, a taxa é muito maior que a obtida usando fibra óptica para uma mesma distância (LIAO *et al.*, 2017).

Já a criptografia de sinais baseada em física clássica pode ser dividida em dois tipos: criptografia dos sinais modulados e criptografia dos sinais em banda-base. Para criptografar sinais modulados, uma técnica bastante utilizada é chamada codificação espectral de fase, (*Spectral Phase Encoding,* SPE) descrita por (CORNEJO, PEREZ e TOCNAYE, 2007). Nela, o sinal a ser encriptado incide em uma grade de difração e é separado em diferentes componentes espectrais. Então uma lente divergente é utilizada para paralelizar a trajetória espacial dessas componentes. A criptografia é realizada por meio de um modulador espacial de luz (S*patial Light Modulador,* SLM), que consegue inserir diferença de fase em conjuntos de componentes espectrais, chamados de fatias. Após o uso do SLM, cada fatia espectral foi acrescida de uma fase diferente e o procedimento contrário é realizado: uma lente convergente é aplicada de modo a reunir todas as fatias em um único ponto, então a grade de difração é utilizada para gerar novamente um único raio, criando o sinal criptografado.

Outra técnica para criptografar sinais modulados é descrita em (ABBADE *et al.*, 2019), por meio da qual dois ou mais sinais são utilizados. O modulador espacial de luz é responsável por permitir ou não a passagem de algumas fatias espectrais do sinal, de modo que as fatias bloqueadas no primeiro sinal não devem ser bloqueadas no segundo. Por fim, as fatias de ambos sinais são multiplexadas e transmitidas por uma rede transparente.

Como é possível notar, a criptografia de sinais modulados utiliza diversos equipamentos, o que faz com que a técnica possua um custo mais elevado. Além disso, a portadora utilizada para o sinal modulado opticamente é um *laser*, cuja frequência central depende da temperatura e pode sofrer flutuações ao longo do tempo. Com isso, a frequência utilizada no transmissor pode não ser a mesma utilizada no receptor, de modo a ocasionar um desalinhamento das fatias espectrais. Para resolver problemas como esse e reduzir o custo da utilização de *hardwar*e adicional, foi proposto fazer a criptografia dos sinais em banda-base utilizando processamento digital de sinais.

Uma técnica de criptografia dos sinais em banda-base foi descrita em (SANTOS, 2020) e leva em consideração SPE e embaralhamento intracanal, ou seja, é realizado um embaralhamento entre as informações do próprio sinal, trocando suas posições. O procedimento foi denominado DSP-SPE-Scr e utiliza técnicas de processamento digital de sinais. O diagrama de blocos para a encriptação está exemplificado na Figura 9.

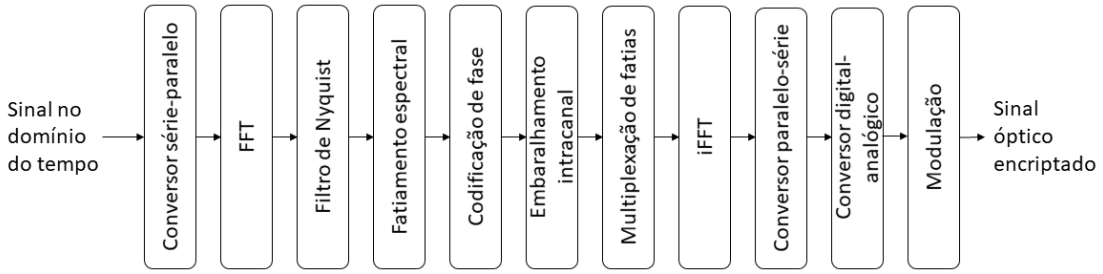

Figura 9 – Diagrama de blocos para o processo de encriptação do DSP-SPE-Scr.

O primeiro passo consiste em aplicar o sinal no domínio do tempo com suas devidas amostras em um conversor série-paralelo, de modo a possibilitar que todas as amostras sejam processadas no mesmo instante de tempo. Em seguida, é realizada uma transformada rápida de Fourier (*Fast Fourier Transform*, FFT) para converter o sinal do domínio do tempo para o domínio da frequência e, uma vez no domínio da frequência, é aplicado um filtro de Nyquist. Após passar pelo filtro, a banda do sinal fica limitada e as amostras ficam com amplitudes semelhantes. Neste momento, são obtidas fatias espectrais similares às utilizadas na SPE, com a diferença que uma fatia espectral compreende várias amostras espectrais. O próximo passo consiste na codificação de fase, ou seja, acrescentar uma diferença de fase em cada fatia espectral. Em seguida, o embaralhamento intracanal é responsável por trocar a posição das fatias dentro do sinal.

Depois, é realizada multiplexação das fatias, seguida de uma FFT inversa, denominada IFTT, para que o sinal retorne ao domínio do tempo. Então, é aplicado um conversor paralelo-série e também um conversor digital-analógico, de modo que o sinal de saída possua vários níveis de amplitude. Por fim, o sinal pode ser modulado da maneira desejada, seja em uma portadora óptica ou até mesmo na frequência de 2,5 GHz para WiFi. No caso apresentado, foi utilizado um sistema óptico.

A transmissão deve ocorrer por uma rede óptica transparente (*Transparent Optical Network,* TON*)*, ou seja, entre transmissor e receptor não podem existir roteadores IP ou qualquer outra estação que processa a informação. Para descriptografar é realizado o processo inverso, adicionando inicialmente um bloco para fazer a sincronização entre receptor e transmissor e também para compensar penalidades impostas pelo meio óptico. Para o DSP-SPE-Scr, todos os procedimentos são realizados por meio de processamento digital de sinal e sem depender de equipamentos como o SLM, o que reduz o custo de sua implementação e flexibiliza a criptografia.

Se cada amostra for considerada uma fatia, é possível obter o maior número possível de mudanças de fase, já que a codificação ocorre em cada fatia. Com o maior número de mudanças de fase, se torna mais complicada a recuperação do sinal por um espião. Com essa alteração, são removidos os blocos de fatiamento espectral e de multiplexação de fatias e o procedimento de encriptação passa a ser ilustrado pela Figura 10. Como este trabalho utiliza uma versão simplificada do DSP-SPE-Scr, observe que o embaralhamento intracanal também é omitido.

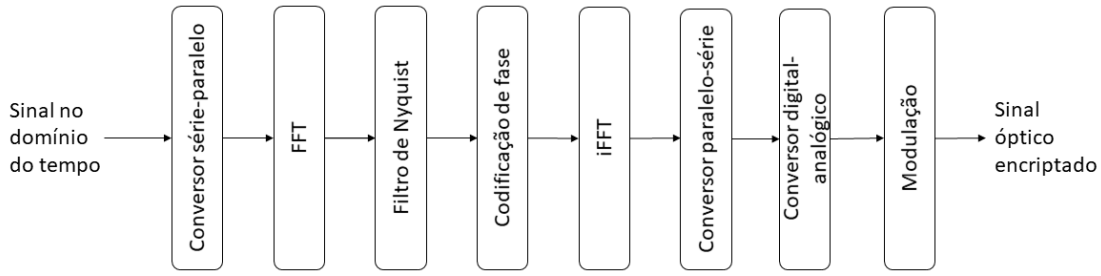

Figura 10 – Diagrama de blocos para a versão simplificada da encriptação do DSP-SPE.

3.2 Proposta de autenticação

Quando o sinal criptografado na camada física é transmitido e chega ao destino, é necessário fazer sua recuperação. Para isso, os instantes corretos de amostragem do sinal devem ser identificados e uma maneira de fazê-lo é adicionar um cabeçalho aos dados transmitidos, chamado de piloto. Para protocolos orientados a bit, recomenda-se a utilização de um delimitador para fazer a separação dos quadros (FOROUZAN, 2008), de modo a destacar onde o sinal criptografado começa e o cabeçalho termina. Considerando os sinais, é possível optar por fazer a sincronização por meio de uma função de correlação entre o sinal obtido no receptor e o sinal utilizado como cabeçalho. Assim, quando a correlação é máxima significa que corresponde exatamente ao cabeçalho e é o momento de iniciar a amostragem para o sinal criptografado.

No entanto, acrescentar o cabeçalho ocasiona maior uso da banda disponível, o que é indesejável pois ocorre desperdício de banda com dados que não representam a informação transmitida de fato. Assim, uma maneira de fazer bom uso do cabeçalho é aproveitá-lo para autenticar e garantir a integridade do sinal, ou seja, com isso o destinatário tem certeza de que o sinal foi realmente enviado pelo remetente e que durante a transmissão ninguém alterou o texto cifrado. A autenticação pode ser obtida aplicando o GCM e utilizando as *tags* como cabeçalho, enquanto integridade é dada pela própria criptografia de sinais, já que a alteração do sinal por um espião acaba alterando também o diagrama de constelação obtido no receptor.

Neste trabalho, a proposta é utilizar o DSP-SPE para fazer a criptografia de sinais e, de maneira paralela, aplicar o GMAC com um AAD e uma chave *k* conhecida por transmissor e receptor. Dessa forma, ambos conseguem calcular a *tag* e o receptor pode comparar a *tag* recebida e a *tag* calculada. Se são iguais, então foi garantida a autenticação do sinal. Se são diferentes, então não é possível garantir que o transmissor é autêntico. Além disso, pode-se notar que somente transmissor e receptor conseguem calcular a *tag*, já que a chave usada por eles é secreta, podendo ser compartilhada por exemplo por meio de QKD. Um exemplo de obtenção da *tag* é ilustrado pela Figura 11, que pode ser adaptado a depender da quantidade de bits do piloto necessários: para *tags* piloto com 32 bits conforme o exemplo, sugere-se gerar quatro *tags* por vez com o GMAC, para *tags* pilotos de 64 bits sugere-se utilizar duas *tags* e assim por diante.

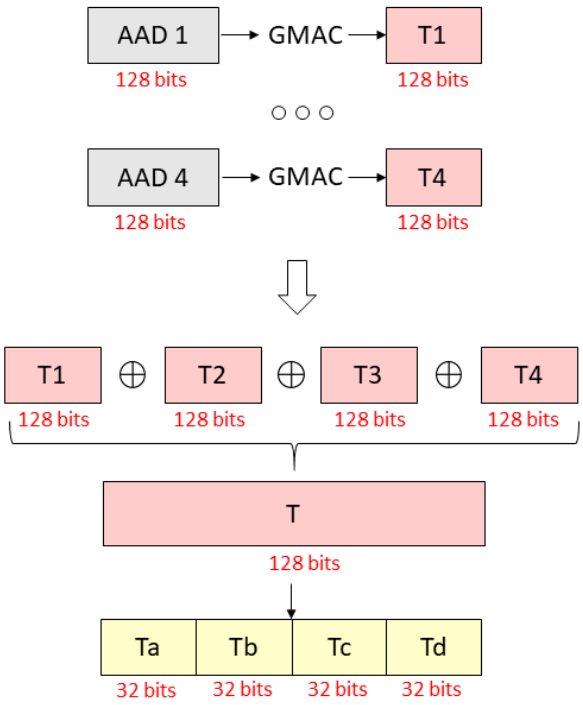

Figura 11 – Proposta de obtenção das *tags* de autenticação.

Para o exemplo da Figura 11 em que a *tag* piloto utilizada possui 32 bits, é aplicado o GMAC para quatro diferentes blocos de AAD, gerando quatro *tags* de 128 bits cada. Em seguida, as *tags* passam por uma operação de XOR entre si, de modo a gerar uma nova *tag* T também com 128 bits. Depois, T é dividida em quatro blocos iguais, cada um com 32 bits de tamanho. Cada novo bloco (T_a, T_b, T_c, T_d) representa uma *tag* piloto a ser utilizada em um novo sinal transmitido e o tamanho foi planejado para consumir a menor banda possível. Neste trabalho, a intenção é verificar possíveis tamanhos para as *tags* piloto e identificar quais possibilitam melhor sincronização e análise do sinal criptografado.

O GMAC também possibilita a obtenção direta de *tags* menores que 128 bits e o procedimento realizado não é muito diferente. Por exemplo, para obter uma *tag* de 32 bits, bastaria aplicar o algoritmo normalmente para obter a *tag* de 128 bits, selecionar os primeiros 32 bits como *tag* e descartar os restantes. No entanto, a geração direta de *tags* menores que 128 bits não é recomendada, já que a probabilidade de um espião escolher aleatoriamente uma *tag* de *n* bits e acertar é de $0{,}5^n$ (DWORKIN, 2007). Assim, com uma *tag* menor é mais provável que o espião consiga encontrá-la. A proposta de

autenticação deste trabalho é feita de modo que todas as *tags* sejam geradas através do GMAC com maior segurança possível, ou seja, com 128 bits de tamanho. Com isso, o XOR é necessário para que os bits da *tag* piloto possam assumir tamanhos menores que 128 bits sem comprometer a segurança, já que o receptor deve utilizar os quatro blocos de piloto recebidos para comparar com a *tag* T calculada.

Um procedimento que pode ser usado para garantir a integridade e fazer com que a autenticidade se torne ainda mais forte seria criptografar parte da *tag* junto com o sinal e utilizar o embaralhamento provido pelo DSP-SPE-Scr. De tal forma, se o sinal criptografado for alterado durante sua transmissão, a *tag* recebida será diferente da *tag* calculada no receptor e o sinal não pode ter sua integridade garantida.

4 – SIMULAÇÕES E RESULTADOS

Este capítulo introduz as técnicas utilizadas na criptografia de dados para manipulação das *tags* e mostra o funcionamento do *software* KryptoSJ, desenvolvido pelo grupo de pesquisa do qual a autora é integrante. O KryptoSJ utiliza programação em Matlab® e conta com blocos que serão descritos ao longo do capítulo. Por motivos de simplicidade, as *tags* referidas nesse capítulo correspondem às *tags* piloto.

Na Seção 4.1 é descrito o algoritmo desenvolvido para a sincronização entre sinal que chega no receptor e *tag*. Também são mostrados os procedimentos para comparação de *tag* recebida e *tag* calculada no receptor. Por fim, a Seção 4.2 mostra os resultados das simulações e algumas discussões pertinentes.

Observa-se que o grupo de pesquisa também desenvolveu um algoritmo similar ao AES que pode ser aplicado à criptografia de sinais (SOUZA, 2021). No entanto, como a técnica ainda está limitada por questões de propriedade intelectual, não foi possível utilizá-la durante esse trabalho. Do ponto de vista de sinais, a DSP-SPE-Scr apresenta valores similares de desempenho ao algoritmo do AES desenvolvido, como por exemplo valores de BER semelhantes e, portanto, as análises feitas durante este trabalho continuariam válidas se fossem aplicadas ao algoritmo do AES. A vantagem da aplicação do AES de camada física seria prover propriedades que não são atendidas pelo DSP-SPE-Scr e a utilização de *tags* nesse contexto pode ser analisada em trabalhos futuros.

4.1 Descrição das simulações

As *tags* utilizadas foram obtidas por meio das simulações em Matlab®, conforme sugerido na proposta de autenticação. Os códigos para o AES e para o GCM foram desenvolvidos anteriormente pela autora durante um projeto de iniciação científica. A obtenção das *tags* por meio das simulações desenvolvidas tem como parâmetros de entrada a chave, um IV que é incrementado e o bloco de informações AAD, sendo todos esses parâmetros compartilhados entre transmissor e receptor. As *tags* são armazenadas para posterior utilização no bloco do transmissor e também no bloco de sincronização, além de ser carregada no receptor para fins de comparação. No transmissor, a função "Mapear" é responsável pela conversão dos bits da *tag* em símbolos e por associá-los aos eixos I e Q a depender da modulação desejada, por exemplo pode-se utilizar a modulação por deslocamento de fase em quadratura (*Quadrature Phase Shift Keying*, QPSK) para

obter 2 bits por símbolo. Depois, o sinal obtido correspondente à *tag* passa pelo filtro de Nyquist definido no algoritmo e é armazenado para utilização no bloco de sincronização.

Concomitantemente, o algoritmo do DSP-SPE-Scr é aplicado para gerar um sinal criptografado. Os códigos estão presentes no *software* desenvolvido pelo nosso grupo de pesquisa, denominado KryptoSJ e que tem variadas versões a depender do tipo de criptografia utilizada, como por exemplo as versões utilizadas e explicadas em (NOGUEIRA, 2022), (SANTOS, 2020) e (SOUZA, 2021). Conforme já indicado, a versão do KryptoSJ utilizada nesse trabalho é baseada principalmente na versão de (SANTOS, 2020) de maneira reduzida e simplificada, ou seja, sem utilizar o embaralhamento (DSP-SPE). Assim, no bloco do transmissor, o sinal é criptografado utilizando os parâmetros de entrada que são compartilhados entre transmissor e receptor, tais como taxa de símbolo, número de símbolos e número de amostras por símbolo. Os valores derivados desses parâmetros também são calculados no código, como a quantidade de amostras, período do símbolo, entre outros. Além disso, junto com os parâmetros de entrada é possível definir se os dados serão ou não criptografados após passarem pelo filtro de Nyquist. Dessa forma, o sinal que desejamos transmitir passa pela criptografia e a *tag* passa pelo processo de mapeamento e pelo filtro de Nyquist, conforme indicado anteriormente. A Figura 12 indica as etapas utilizadas, com as duas opções de blocos para o transmissor, o canal, o bloco de sincronização e as duas opções do receptor.

No bloco de sincronização, que deve ser utilizado no receptor, foram carregadas como variáveis de entrada as informações obtidas para a *tag* e para o sinal criptografado. Em seguida, a *tag* e o sinal foram concatenados no domínio do tempo, gerando uma nova variável a ser analisada. O próximo passo consistiu em realizar a sobreamostragem (*oversampling)* da nova variável, ou seja, adicionar amostras de modo a simular de forma mais precisa um sinal contínuo. Para isso, nas variáveis de entrada definiu-se o fator de sobreamostragem (f_{ss}) e, consequentemente, um novo período de amostragem (T_s). Por exemplo, se o período de amostragem inicial é definido por T_{sy}, então T_s é dado pela divisão do período de amostragem inicial pelo fator de sobreamostragem. Neste trabalho adotou-se um fator de sobreamostragem igual a 16, de modo que a cada intervalo entre duas amostras iniciais foram adicionadas mais 15 amostras. Para obter os valores de amplitudes correspondentes às novas amostras utilizou-se a função de interpolação em uma dimensão presente no Matlab®. A interpolação foi realizada para a parte real e para

a parte imaginária do sinal concatenado, de maneira paralela. Depois, foi feita a sobreamostragem somente da *tag*, de maneira que o receptor consiga utilizá-la para fazer a comparação com o sinal concatenado que foi recebido.

Em seguida, foram adicionados zeros no início e no final em quantidade definida pelo valor de f_{ss}, ou seja, foram adicionados 16 zeros no início e 16 zeros no final do sinal concatenado e já sobreamostrado. Essa estratégia foi adotada para simular a possibilidade de o receptor iniciar o processo de amostragem em algum instante diferente do valor adequado. Como por exemplo na Figura 13, o instante correto é definido pelas amostras em vermelho e o receptor inicia seu processo de amostragem somente 3 amostras depois. Observe que o exemplo conta com $f_{ss} = 8$ para simplificação. Então, para simular o processo de amostragem iniciado no valor diferente do ideal, foi selecionado uma variável pseudo-aleatória "n_rand", cujos valores podem assumir qualquer valor na faixa entre 0 e $f_{ss} - 1$, de modo a representar o atraso no processo. No exemplo da Figura 13, essa variável é igual a 3.

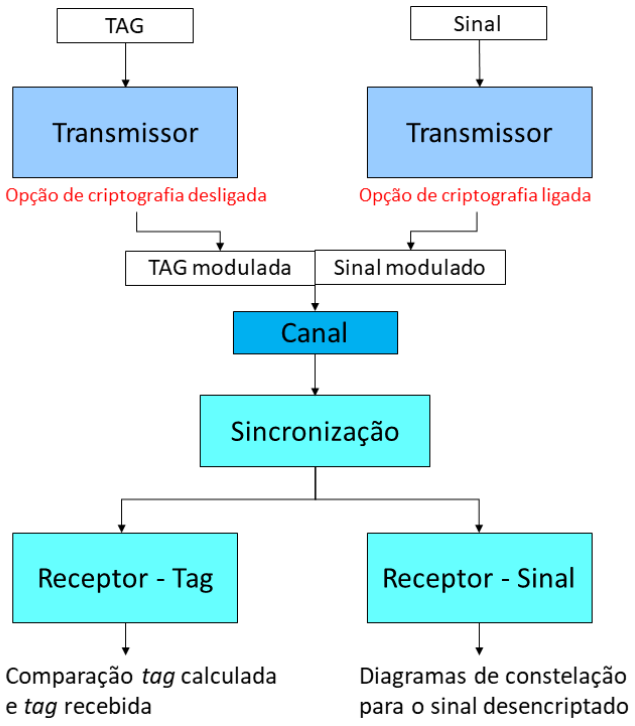

Figura 12 – Diagrama de blocos das simulações utilizadas.

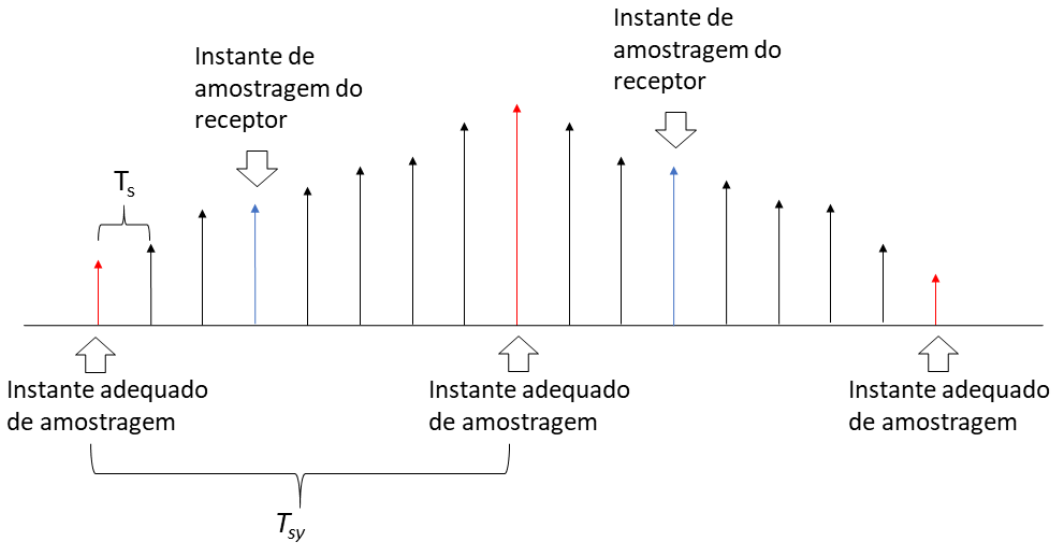

Figura 13 – Exemplo de instante de amostragem adequado e instante de amostragem do receptor.

Assim, o vetor que chega de fato no receptor é dado pelos instantes de amostragem deslocados de “n_rand” posições. Para que o tempo não se altere, o uso de zeros antes e depois é justificado: como a função utilizada faz o deslocamento das amostras de maneira circular, ou seja, a última amostra vai para a primeira posição, então ao deslocar *n* amostras para a direita deslocam-se também *n* zeros para a primeira posição, possibilitando a leitura correta dos momentos sem alteração do tempo.

Por conta desse deslocamento, assume-se que o receptor não conhece a posição em que a *tag* é iniciada. Para descobri-la, o algoritmo desenvolvido utiliza uma convolução entre o vetor deslocado e a *tag* sobreamostrada, já que o receptor tem condições de calcular a tag e fazer sua sobreamostragem. Como a convolução representa a área obtida quando dois sinais se sobrepõem no domínio do tempo, então é possível assumir que o seu ponto máximo nesse caso ocorre quando os dois sinais que estão sendo comparados são iguais, ou seja, exatamente no momento em que a *tag* é iniciada. Portanto, o algoritmo armazena a amostra em que a convolução é máxima e sua posição correspondente. Em seguida, para obter o instante correto no vetor de tempo, a quantidade de amostras correspondentes ao tamanho da *tag* deve ser subtraída da posição correspondente, já que a função de convolução do Matlab® gera um vetor de tamanho dado pela soma dos tamanhos de seus vetores de entrada e subtraindo 1. Assim, como a posição desejada é no vetor deslocado, deve-se subtrair a quantidade de amostras correspondentes ao tamanho da *tag*. No diagrama sugerido, o bloco de sincronização vem

depois do canal, que é o responsável por adicionar ruído cuja amplitude pode ser modificada nos parâmetros de entrada do sistema.

Por fim, com as informações do instante correto é possível amostrar o sinal e obter sua versão em tempo discreto: amostra-se inicialmente na posição desejada e depois de f_{ss} em f_{ss} amostras, de modo a remover toda a sobreamostragem e obter a quantidade de amostras inicial. De posse do sinal recuperado em tempo discreto é possível separar as amostras correspondentes à *tag* das amostras correspondentes ao sinal criptografado, já que o tempo correspondente à *tag* é conhecido pelo receptor. Assim, as duas variáveis são armazenadas e cada uma segue para um bloco diferente dentro do receptor, conforme orientado pela Figura 12.

No primeiro bloco do receptor é realizada a análise e comparação da *tag* recebida com a *tag* calculada. Para isso, utiliza-se parte do *software* KryptoSJ de modo a converter as amplitudes dos símbolos da *tag* em bits por meio da função chamada "Demapear", que faz operação contrária à função "Mapear". Então, é realizada uma operação de XOR entre as duas partes da *tag* em bits, tanto no eixo I quanto no eixo Q: bit 1 resultante indica que houve diferença entre as *tags* e bit 0 resultante mostra que as *tags* são iguais. Para facilitar a análise, foi mostrada na janela de comandos a informação "Não ocorreram erros" e "Ocorreu algum erro" para cada um dos eixos. Já o segundo bloco serve para fazer a desencriptação do sinal, recuperando as componentes espectrais por meio da chave que foi compartilhada por receptor e transmissor e identificando de forma gráfica os diagramas de constelação obtidos antes e após a desencriptação, para fins de comparação. Também é calculada a taxa de erro de bit (*Bit Error Rate,* BER) para cada sinal desencriptado, utilizando o *software* com as técnicas descritas em (SANTOS, 2020) e (NOGUEIRA, 2022).

4.2 Apresentação dos resultados

Inicialmente a *tag* passa pelo bloco do transmissor. Para os resultados a seguir, os parâmetros utilizados incluíram: *tag* de 64 bits no total, resultando em 32 símbolos; 2 amostras por símbolo para gerar um sinal QPSK; taxa de símbolo de 28GBaud, fator de decaimento do filtro (*roll-off)* de 0,02 e adição de ruído aditivo, gaussiano e branco (*Additive White Gaussian Noise*, AWGN). Como a *tag* não deve ser criptografada, a opção de criptografia de sinais foi desativada. Para o sinal, a criptografia foi ativada e os

parâmetros foram quase iguais, com diferença na utilização de 128 símbolos ao invés dos 32 símbolos da *tag*.

No bloco de sincronização é possível analisar a variável composta da concatenação de *tag* e sinal, que possui como quantidade de símbolos a soma dos símbolos da *tag* e do sinal criptografado, ou seja, 160 símbolos. A Figura 15 indica a variação das amplitudes do vetor concatenado em relação ao tempo. Os gráficos da esquerda mostram a parte real do sinal e os da direita mostram a parte imaginária. Além disso, a primeira linha mostra o sinal em tempo discreto, a segunda linha mostra o sinal sobreamostrado e a terceira linha mostra o sinal sobreamostrado com os f_{ss} zeros adicionados no início e no final. Para melhor visualização foram apresentadas somente as amostras iniciais na Figura 14 e as amostras finais foram indicas na Figura 15. Em seguida, é realizada a sobreamostragem somente da *tag*, indicada pelos dois gráficos na segunda linha da Figura 16. A primeira linha indica a *tag* como sinal discreto. Além disso, as amostras da *tag* correspondem às amostras iniciais das Figuras 14 e 15.

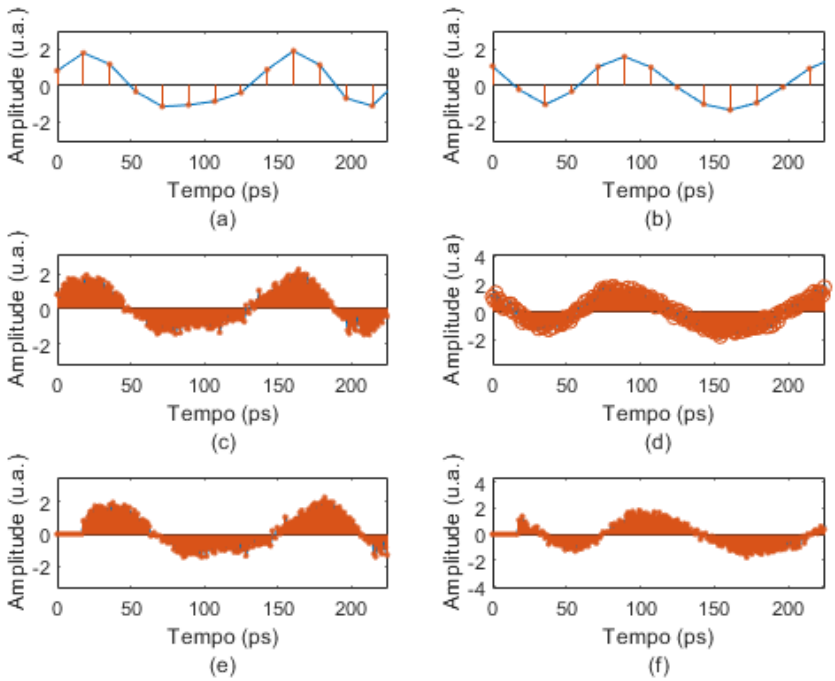

Figura 14 – Gráficos da amplitude em relação ao tempo, amostras iniciais para (a) parte real do sinal concatenado em tempo discreto, (b) parte imaginária do sinal concatenado em tempo discreto, (c) parte real do sinal sobreamostrado, (d) parte imaginária do sinal sobreamostrado, (e) parte real do sinal sobreamostrado com adição de zeros, (f) parte imaginária do sinal sobreamostrado com adição de zeros.

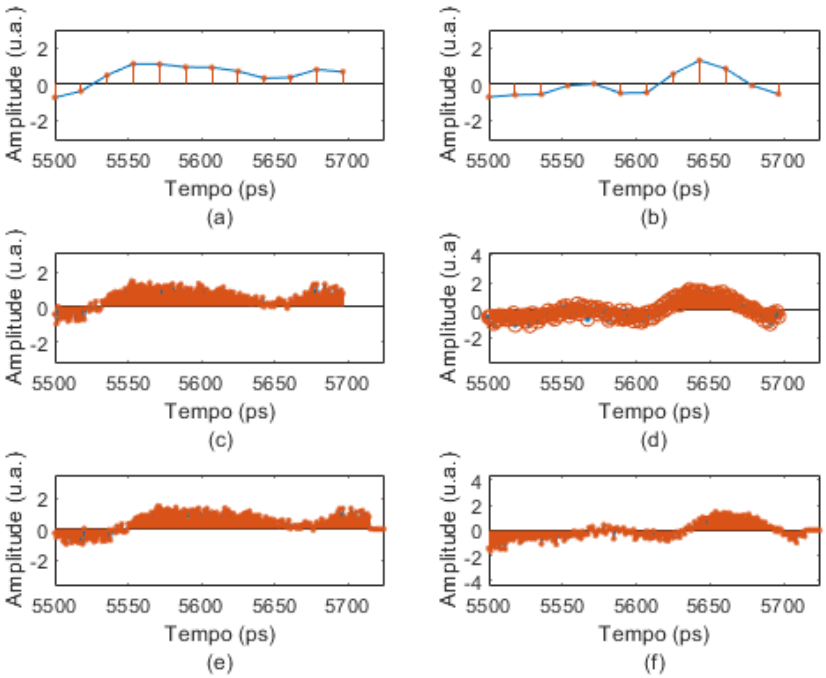

Figura 15 – Gráficos da amplitude em relação ao tempo, amostras finais para (a) parte real do sinal concatenado em tempo discreto, (b) parte imaginária do sinal concatenado em tempo discreto, (c) parte real do sinal sobreamostrado, (d) parte imaginária do sinal sobreamostrado, (e) parte real do sinal sobreamostrado com adição de zeros, (f) parte imaginária do sinal sobreamostrado com adição de zeros.

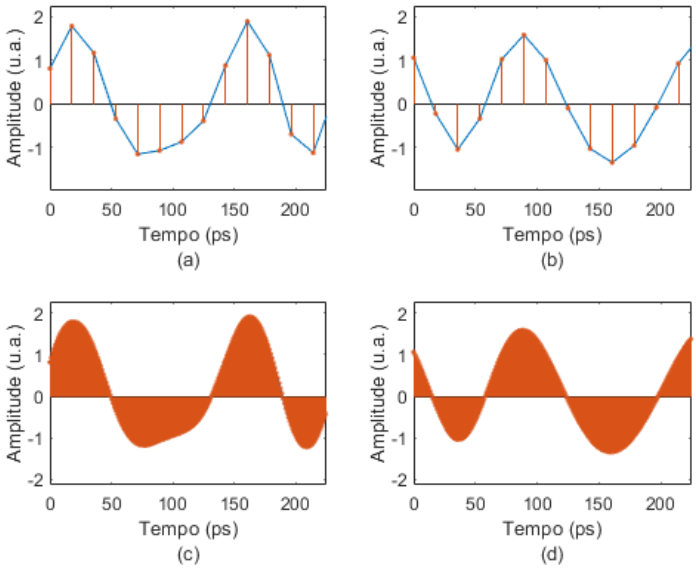

Figura 16 – Gráficos da amplitude em relação ao tempo para (a) parte real da *tag* em tempo discreto, (b) parte imaginária da *tag* em tempo discreto, (c) parte real da *tag* sobreamostrada, (d) parte imaginária da *tag* sobreamostrada.

Depois, realizou-se a simulação do deslocamento no tempo e a identificação do momento correto de amostragem. Para comparar o sinal recuperado e o sinal antes do deslocamento, denominado sinal original, foram plotados os gráficos da Figura 17. Nela,

o primeiro gráfico representa a parte real do sinal e o segundo mostra a parte imaginária, a curva em azul identifica o sinal recuperado e a curva em laranja mostra o sinal original. Observe que para o nível de ruído adicionado o sinal foi recuperado com sucesso.

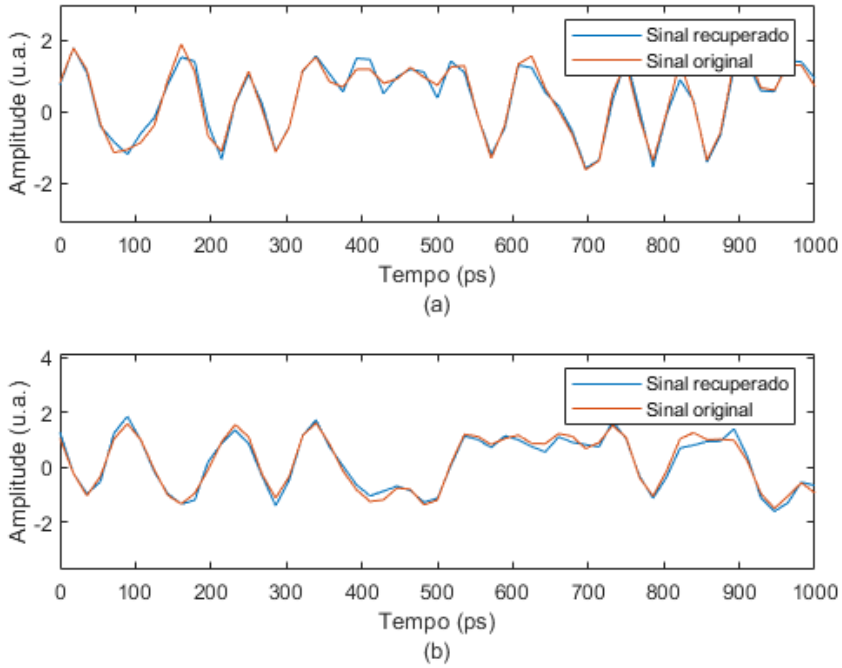

Figura 17 – Gráficos da amplitude em relação ao tempo para (a) parte real do sinal concatenado recuperado e do sinal original, (b) parte imaginária do sinal concatenado recuperado e do sinal original.

Em seguida, foram separadas as partes correspondentes à *tag* e ao sinal criptografado. A Figura 18 indica a *tag* resultante na saída do bloco de sincronização, com suas representações em tempo e em frequência. Para o espectro da amplitude a banda é limitada em aproximadamente 14 GHz pois passou pelo filtro de Nyquist definido no transmissor com *roll-off* de 0,02. Nos momentos em que o espectro da amplitude tende a zero, o espectro de fase é indeterminado e, conforme mostrado na Figura 18, pode assumir qualquer valor. Já a parte correspondente ao sinal criptografado está indicada na Figura 19. Observe que a banda é limitada da mesma maneira que na *tag*, pois o sinal também passou pelo filtro.

Em seguida, foi utilizado o primeiro bloco do receptor para fazer a comparação da *tag* calculada e da *tag* recebida. Para esta simulação, as *tags* não apresentaram diferença entre si, indicando a autenticidade da mensagem enviada. No entanto, acrescentar muito ruído pode ocasionar alteração da *tag* mesmo com a presença de um filtro no receptor.

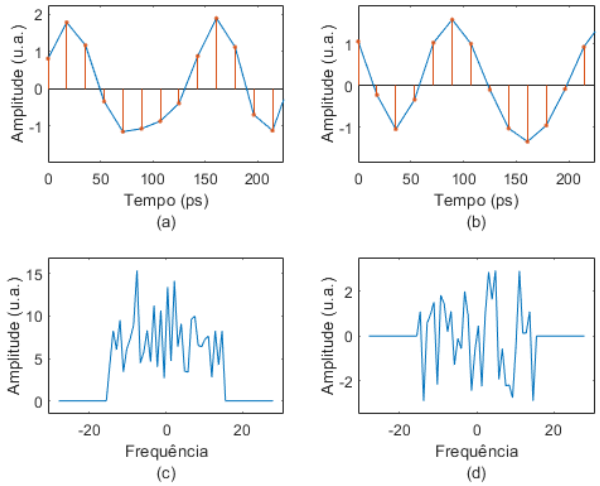

Figura 18 – Representação da *Tag* resultante do bloco de sincronização para (a) parte real no domínio do tempo, (b) parte imaginária no domínio do tempo, (c) espectro da amplitude, (d) espectro da fase.

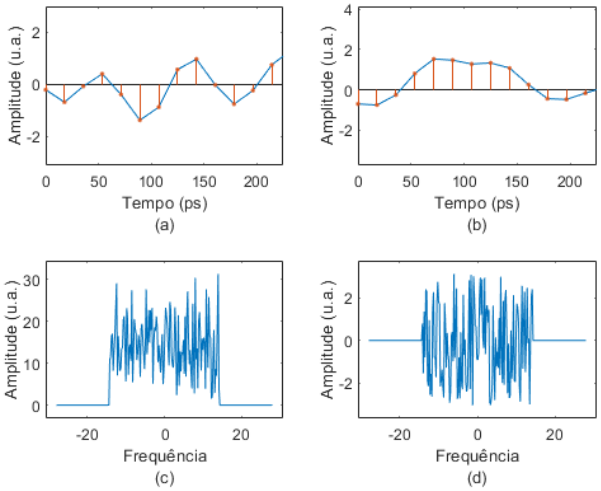

Figura 19 – Representação em tempo e frequência do sinal criptografado resultante do bloco de sincronização para (a) parte real no domínio do tempo, (b) parte imaginária no domínio do tempo, (c) espectro da amplitude do sinal, (d) espectro da fase do sinal.

Já os diagramas de constelação foram obtidos antes da criptografia no transmissor, depois da criptografia e da adição de ruído e, por último, para o sinal recuperado e desencriptado. Como é possível observar, a criptografia utilizada pelo *software* DSP-SPE-Scr que é indicada na Figura 20(b) gera grande alteração da constelação, de modo que o sinal só pode ser recuperado fazendo a desencriptação com a chave correta. Na Figura 20(c) o sinal é desencriptado com a chave correta mas a constelação não é recuperada totalmente, já que alterações no instante de amostragem e a adição de ruído provavelmente alteraram o sinal. Na Figura 21, é possível notar como a adição de mais ruído degrada o diagrama de constelação.

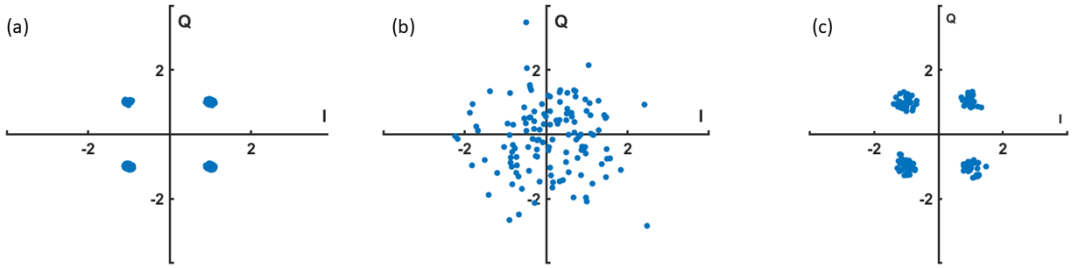

Figura 20 – Diagramas de constelação definidos (a) antes da criptografia, (b) depois da criptografia e da adição de ruído, (c) depois da recuperação e desencriptação.

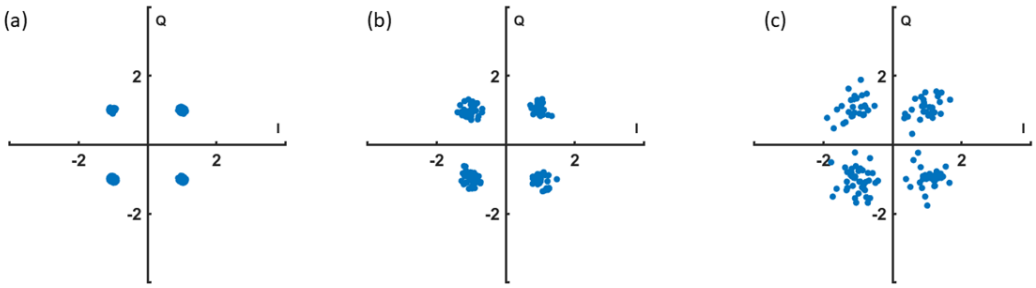

Figura 21 – Diagramas de constelação definidos depois da desencriptação para: (a) SNR = ∞, (b) SNR = 19,9 dB e (c) SNR = 10,4 dB.

Por fim, para realizar a comparação da BER com diferentes tamanhos de *tags* é necessário a análise de vários valores. Para cada tamanho de *tag* variando de 32, 64, 96 e 128 bits foram analisados 128 sinais, cada um com 128 símbolos. A BER para cada sinal relacionado aos diferentes tamanhos de *tag* foi armazenada e, posteriormente, foi feita a BER média para cada tamanho de *tag.* Com isso, foi possível plotar a variação da BER conforme indicado na Figura 22. As diferentes curvas representam níveis de razão sinal-ruído (S*ignal-to-Noise Ratio,* SNR), que são menores para maiores valores de BER. Em relação aos tamanhos da *tag* é possível notar pouca variação, indicando BER praticamente constante. Portanto, a adição de mais ruído altera a BER, mas não faz diferença utilizar *tags* maiores ou menores para fazer a sincronização.

Já a Figura 23 mostra a variação da BER em função do aumento da SNR, considerando cada tamanho de *tag* apresentado. É possível notar que as curvas são bastante similares, variando de forma bem sutil somente quando a SNR é maior que 22 dB. Assim, a similaridade das curvas obtidas indica também que a alteração do tamanho da *tag* não altera a BER do sinal no receptor.

Outro parâmetro que pode ser usado para compreender o tamanho ideal da *tag* é o *overhead*, definido pela divisão entre quantidade de bits da *tag* e quantidade de bits do sinal. Quanto maior *overhead*, maior a quantidade de bits da *tag* em relação aos bits do sinal. Portanto, para utilizar menor banda e a menor relação, é desejável o uso da menor *tag* possível. Como a análise da BER indicou que os 32 bits são suficientes para a sincronização, este é o tamanho da *tag* ideal analisado.

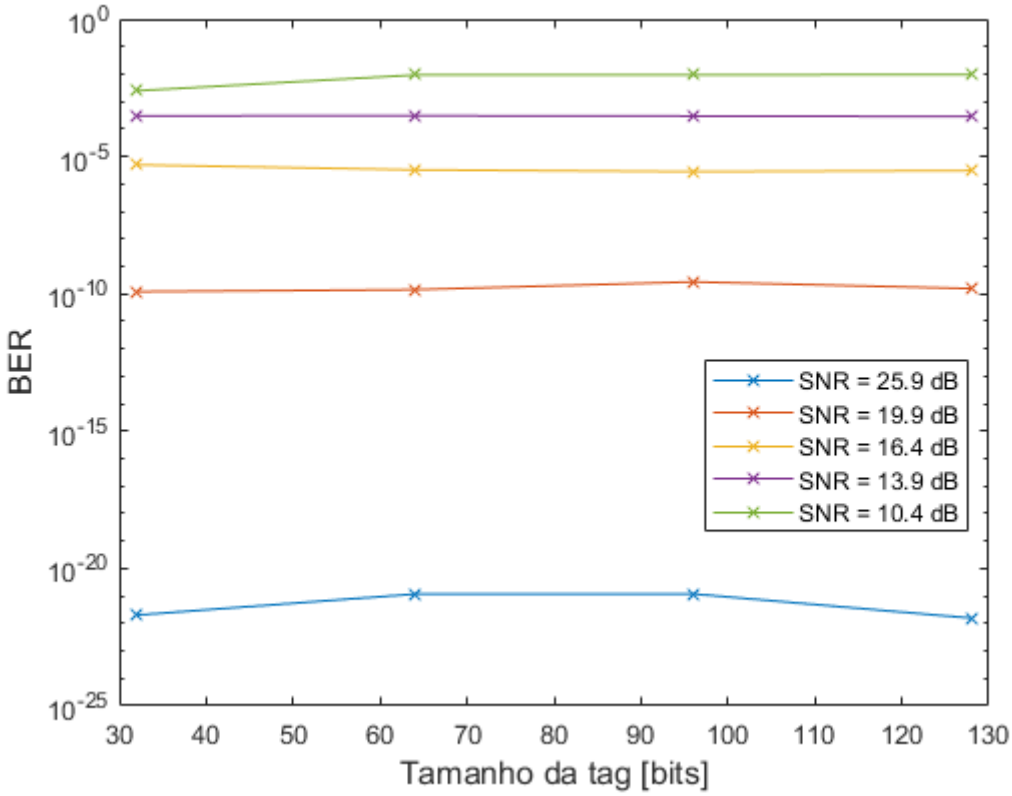

Figura 22 – Variação da BER em função do tamanho da *tag* utilizada.

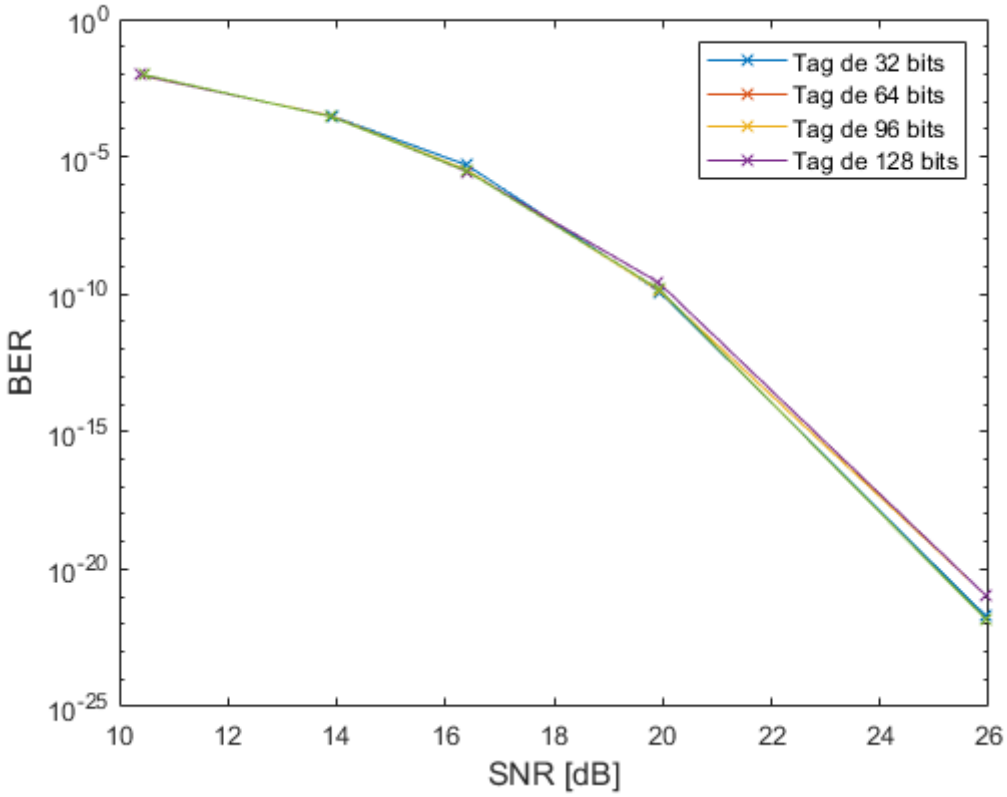

Figura 23 – Variação da BER em função da SNR.

5 – CONCLUSÕES

Neste trabalho foram abordadas com sucesso técnicas de criptografia de dados e criptografia de sinais de forma paralela, com o propósito de fazer autenticação e identificar o correto momento de amostragem de um sinal, garantindo a segurança das informações que trafegam na camada física. Na criptografia de dados foi utilizado um modo de operação do AES, o GCM, que possibilitou a obtenção de *tags* que somente podem ser calculadas por indivíduos que compartilham uma chave *k*. De tal modo, o transmissor concatena a *tag* ao sinal criptografado que deseja enviar e o receptor garante a autenticidade da mensagem. Para a criptografia de sinais foi utilizada uma técnica de criptografia de sinais em banda-base, a DSP-SPE-Scr, com modulação QPSK.

Foi empregado com sucesso um algoritmo para fazer a concatenação do sinal criptografado já modulado e da *tag* modulada. A correlação entre os sinais mostrou-se uma ferramenta útil para a detecção do exato momento de amostragem para o caso simulado em que o receptor começa a amostragem com um atraso aleatório. A comparação entre o sinal recuperado e o sinal original indicou que o momento adequado de amostragem foi identificado de forma razoável.

Foi considerada também a necessidade de sobreamostrar os sinais para simular um sinal em tempo contínuo, de modo que a detecção do correto instante de amostragem se tornasse mais precisa. Para as simulações obtidas, foi possível notar uma alteração abrupta nos instantes em que se adicionou zeros, situação que pode ser resolvida com a realização da interpolação após essa adição e não antes, como realizado neste trabalho. Tal alteração e a análise de suas implicações podem ser discutidas em trabalhos futuros

Já o fato de o receptor calcular e, portanto, conhecer a *tag* modulada e sobreamostrada possibilitou sua separação do sinal criptografado. A comparação da *tag* calculada e da *tag* recebida mostrou que na maioria dos casos o receptor consegue constatar a autenticidade da mensagem se aplicar um filtro simples na *tag* recebida, com exceção para os casos em que o ruído degradou muito o sinal.

Por fim, o sinal identificado no receptor passou pela desencriptação e pelo filtro com sucesso. Com a análise da BER em função do tamanho das *tags*, notou-se que alterações de tamanho são indiferentes para este caso. Adicionar mais ruído também não faz com que as *tags* de tamanho menor tenham desempenho pior que as *tags* de tamanho

maior para a faixa de ruído considerada. Portanto, a *tag* mais indicada para fazer a sincronização e garantir a autenticidade do sinal transmitido é a de 32 bits, já que apresenta *overhead* baixo e utiliza a menor quantidade de banda possível.

REFERÊNCIAS

ABBADE, M. L. F. et al. **Security in Optical Communication Systems:** Data Encryption and Beyond. 2021 SBFoton International Optics and Photonics Conference (SBFoton IOPC). [S.l.]: [s.n.]. 2021. p. 1-6.

ABBADE, M. L. F. et al. All-Optical Encryption Using Multi-Channel Spectral Shuffling. **IEEE Photonics Technology Letters**, v. 31, n. 1, p. 98-101, Janeiro 2019.

BENNET, C. H.; BRASSARD, G. **Quantum Cryptography:** Public Key Distribution and Coin Tossing. International Conferece on Computers, Systems & Signal Processing. Bangalore: [s.n.]. 1984. p. 175-179.

CORNEJO, J. ; PEREZ, ; TOCNAYE, J.-L. D. B. D. L. WDM-Compatible Channel Scrambling for Secure High-Data-Rate Optical Transmissions. **Journal of Lightwave Technology**, v. 25, n. 8, p. 2081-2089, Agosto 2007.

DAEMEN, J.; RIJMEN, V. **The Design of Rijndael**. 1ª. ed. Heidelberg: Springer Berlin, 2002.

DWORKIN, M. **NIST Special Publication 800-38A: Recommendation for block cipher modes of operation**. National Inst of Standards and Technology. Gaithersburg. 2001.

DWORKIN, M. **NIST Special Publication 800-38D: Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC**. National Institute of Standards and Technology. Gaithersburg. 2007.

EKERT, A. K. Quantum Cryptography Based on Bell's Theorem. **Physical Review Letters**, v. 67, n. 6, p. 661-663, Agosto 1991.

FOROUZAN, B. A. **Comunicação de Dados e Redes de Computadores**. 4ª. ed. São Paulo: McGraw-Hill, 2008.

GUERON, S.; KOUNAVIS, M. E. **Intel® carry-less multiplication instruction and its usage for computing the GCM mode.** Intel Corporation. [S.l.]. 2010.

LIAO, S. et al. Satellite-to-ground quantum key distribution. **Nature**, n. 549, p. 43-47, Agosto 2017.

LUCAMARINI, M. et al. Overcoming the rate–distance limit of quantum key distribution without quantum repeaters. **Nature**, v. 557, p. 400-403, Maio 2018.

MCGREW, D.; VIEGA, J. **The Galois/counter mode of operation (GCM)**. Submissão para o Nacional Institute of Standards and Technology. [S.l.]. 2004.

NOGUEIRA, M. P. **Criptografia física por embaralhamento espectral aplicada a sinais**. Universidade Estadual Paulista Júlio de Mesquita Filho, Faculdade de Engenharia. São João da Boa Vista, p. 65. 2022.

PAAR, C.; PELZL, J. **Understanding Cryptography:** A Textbook for Students and Practitioners. 1. ed. Londres: Springer, 2010.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. **Communications of the ACM**, v. 21, n. 2, p. 120-126, Fevereiro 1978.

SANTOS, M. D. O. **Criptografia na camada física baseada em codificação espectral implantada por meio de DSP e aplicada a redes ópticas**. Universidade Estadual Paulista Júlio de Mesquita Filho - Câmpus de São João da Boa Vista. São João da Boa Vista, p. 1-65. 2020.

SOUZA, W. S. **Adaptação de criptografia espectral ao paradigma do "Advanced Encryption Standard"**. Universidade Estadual Paulista Júlio de Mesquita Filho - Câmpus de São João da Boa Vista. São João da Boa Vista. 2021.

UEMATSU, T. et al. Design of a Temporary Optical Coupler Using Fiber Bending for Traffic Monitoring. **IEEE Photonics Journal**, v. 9, n. 6, Dezembro 2017.

YANG, X. et al. Chaotic Encryption Algorithm Against Chosen-Plaintext Attacks in Optical OFDM Transmission. **IEEE Photonics Technology Letters**, v. 28, p. 2499-2502, Novembro 2016.