
UNIVERSIDADE D
COIMBRA

Pedro Miguel de Sousa Martins

# Implementação de um modelo de criptografia em códigos gráficos bidimensionais

## Integração dos códigos QR-code e UniQode

**Dissertação no âmbito do Mestrado Integrado em Engenharia Electrotécnica e de Computadores, ramo de especialização em Automação orientada pelo Professor Doutor Nuno Miguel Mendonça da Silva Gonçalves e apresentada à Faculdade de Ciências e Tecnologia da Universidade de Coimbra no Departamento de Engenharia Electrotécnica e de Computadores.**

Abril de 2021

UNIVERSIDADE D
COIMBRA

# Implementação de um modelo de criptografia em códigos gráficos bidimensionais

Integração dos códigos QR-code e UniQode

**Pedro Miguel de Sousa Martins**

Dissertação para obtenção do Grau de Mestre em
**Engenharia Electrotécnica e de Computadores**

Orientador: Doutor Nuno Miguel Mendonça da Silva Gonçalves

**Júri**

| | |
|---|---|
| Presidente: | Doutor Gabriel Falcão Paiva Fernandes |
| Vogal: | Doutor Marco Alexandre Cravo Gomes |
| Vogal: | Doutor Nuno Miguel Mendonça da Silva Gonçalves |

**Abril de 2021**

- À minha mãe

# Agradecimentos

Quero agradecer ao meu orientador, ao professor Nuno Gonçalves, pelo seu suporte e a oportunidade de finalizar esta dissertação, assim como, pelo seu conhecimento e experiência fornecida.

Gostaria também de agradecer à minha mãe, irmã e familiares pelo incentivo, apoio e compreensão que me proporcionaram ao longo deste anos.

Por último aos meus colegas de casa por todas as memórias vividas e por todo o apoio prestado durante o processo de escrita e revisão desta dissertação.

A todos,
*Muito Obrigado*

# Resumo

A revolução tecnológica permite aos consumidores a compra de um determinado produto através da conectividade à Internet. Deste modo qualquer empresa procura uma forma de publicitar estes produtos através de uma plataforma *web* ou uma aplicação para telemóvel, para que qualquer consumidor possa obter facilmente informação do mesmo de uma forma eficiente. Além da importância da divulgação em massa de um dado produto, uma das maiores preocupações das empresas é garantir a autenticidade e a validação desse produto para que seja evitada a contrafação do mesmo. Caso este fenómeno não seja devidamente tratado com mecanismos corretos pode acabar por gerar prejuízos elevados às empresas.

Uma possível solução para garantir não só a identificação mais rápida de produtos por sistema automáticos, como garantir a autenticidade e a segurança do mesmo, são os códigos de leitura óticos, como por exemplo os códigos de barras ou Quick Response code (QR-code). Este último, o QR-code, é designado como uma das maiores referências em termos de divulgação e marketing de produtos. Estes códigos gráficos bidimensionais apresentam grande utilidade de armazenamento de informação sobre um determinado produto e permitem ao consumidor aceder à informação do mesmo através de uma forma bastante eficiente.

Para combater a contrafação e contrariar o crescimento da mesma, surgiu um novo código gráfico bidimensional, designado por UniQode®, que resultou de um trabalho de investigação por parte de um grupo de investigadores do Instituto de Sistemas e Robótica da Universidade de Coimbra (ISR) em parceria com a Imprensa Nacional-Casa da Moeda (INCM).

Nesta dissertação aliando a tecnologia UniQode® a métodos criptográficos, como por exemplo a criptografia simétrica e assimétrica, foi desenvolvida e definida uma nova metodologia para que esta criptografia híbrida no UniQode® pudesse ser integrada e embutida num dado QR-code. Para prova de conceito desta metodologia, foi adaptada uma aplicação em iOS onde se consegue com sucesso a leitura deste código leitura ótico.

# Palavras Chave

QR-code, UniQode, Criptografia simétrica, Criptografia assimétrica, Códigos gráficos bidimensionais, Códigos de leitura ótica

# Abstract

The technological revolution allows consumers to purchase a particular product through Internet connectivity. In this way, any company is looking for a way to advertise these products through a *web* platform or a mobile application, so that any consumer can easily obtain information about it in an efficient way. In addition to the importance of mass dissemination of a given product, one of the biggest concerns of companies is to guarantee the authenticity and validation of that product in order to avoid counterfeiting. If this phenomenon is not properly dealt with the correct mechanisms, it can end up causing high losses to companies.

A possible solution to guarantee not only the faster identification of products by automatic systems, but also to guarantee its authenticity and security, are machine readable codes, such as bar codes or QR-code. The latter, QR-code, is designated as one of the biggest references in terms of product dissemination and marketing. These two-dimensional graphic codes are very useful for storing information about a certain product and allowing the consumer to access the information of the same through a very efficient way.

To combat counterfeiting and counteract its growth, a new two-dimensional graphic code, called UniQode®, emerged, which resulted from the research work by a group of researchers from ISR in partnership with INCM.

In this dissertation, combining UniQode® technology with cryptographic methods, such as symmetric and asymmetric cryptography, a new methodology was developed and defined so that this hybrid encryption in UniQode® could be integrated and embedded in a given QR-code. For proof of concept of this methodology, an iOS application was adapted where the reading of this machine readable code is successfully achieved.

# Keywords

QR-code, UniQode, Symmetric cryptography, Asymmetric cryptography, Two-dimensional graphic codes, Machine Readable Codes

# Índice

# Lista de Figuras

# Lista de Tabelas

# Lista de Acrónimos

**QR-code** Quick Response code

**ISR** Instituto de Sistemas e Robótica da Universidade de Coimbra

**INCM** Imprensa Nacional-Casa da Moeda

**AES** Advanced Encryption Standard

**RSA** Rivest–Shamir–Adleman

**LSB** Least Significant Bit

**MRC** Machine Readable Codes

**DUA** Documento Único Automóvel

**OAEP** Optimal Asymmetric Encryption Padding

**DES** Data Encryption Standard

**NIST** Instituto Nacional de Padrões e Tecnologia

# 1

# Introdução

## Conteúdo

## 1.1 Contexto

Desde os tempos da revolução industrial até à atual revolução tecnológica verificou-se um desenvolvimento exponencial dos mercados. Estes originaram uma maior competição entre as empresas na comercialização dos seus produtos.

Nos dias de hoje, e com a elevada densidade de dispositivos conectados à internet é imperativo que as empresas desenvolvam os seus produtos com a preocupação de que a informação sobre os mesmos possa ser facilmente acedida em qualquer instante na internet por parte dos seus consumidores.

Códigos de leitura ótica surgem assim como uma das possíveis soluções para a identificação mais rápida de produtos por sistemas automáticos. Apesar de serem muito comuns, estes códigos de leitura ótica são cada vez mais vitais para interações, transações e marketing. São referências essenciais para negócios de logística, contabilidade, supermercados, armazéns e muito mais, onde apresentam uma grande utilidade para identificação de produtos e armazenamento das características de um determinado produto. Estes códigos de leitura ótica podem ser unidimensionais ou bidimensionais como o caso do Quick Response code (QR-code).

A tecnologia designada como QR-code [1], permite às empresas, através de um código gráfico com 40 versões diferentes, uma maior capacidade de armazenar informação do respetivo produto. Esta tecnologia é usada em diversas plataformas web ou aplicações de telemóvel, o que devido ao seu fácil acesso leva a uma maior interação entre produto-consumidor, gerando um número maior de vendas do mesmo.

No entanto, apesar do QR-code ser uma tecnologia popular ao nível de marketing, é necessário garantir a segurança e confidencialidade da informação acerca do produto, procurando assim evitar a contrafacção dos mesmos.

Atualmente, a criptografia é um dos métodos mais robustos e utilizados na segurança de informação. Esta pode ser simétrica ou assimétrica, onde a diferença reside na lógica de usar a mesma chave para encriptar ou desencriptar ou usar chaves diferentes para cifrar a informação. Métodos criptográficos podem ser aplicados a códigos de leitura ótica, tornando mais difícil a extracção de informação por parte de terceiros e a geração de novos códigos válidos. Desta forma, os códigos de leitura ótica que utilizem criptografia são mais seguros quer na geração, quer na descodificação, e por isso, permitem garantir de uma forma mais segura a autenticidade dos produtos.

Recentemente um grupo de investigadores do Instituto de Sistemas e Robótica da Universidade de Coimbra (ISR) em parceria com a Imprensa Nacional-Casa da

Moeda (INCM) desenvolveu um novo código gráfico bidimensional, designado por UniQode®. Esta tecnologia garante não só o armazenamento dos dados como a segurança dos mesmos. A tecnologia UniQode® através de diferentes camadas de segurança permite uma maior facilidade de validação de uma marca ou produto aumentando assim a dificuldade de criação de novos códigos legítimos, permitindo também, tal como o QR-code, o armazenamento de informação de uma forma segura e a posterior identificação do produto.

## 1.2 Motivação

Ao longo dos últimos tempos, é possível constatar a massificação de dispositivos ligados à internet. Com este fenómeno os mercados adaptaram as suas estratégias para promover os seus produtos. No entanto esta preocupação com a conectividade levanta questões como a segurança e a autenticidade do produto, que terão de ser garantidas.

Contudo a dificuldade de assegurar estes mecanismos resulta num outro fenómeno, designado por contrafação. Esta tem vindo a aumentar por exemplo em setores industriais, como na indústria têxtil, ou na indústria tabaqueira, representando perdas significativas para as empresas. De notar, que neste último ano uma das áreas que mais procura garantir a autenticidade e validação do seu produto e evitando assim a sua contrafação, é a área da saúde em geral e em particular a indústria farmacêutica.

Uma das soluções viáveis que pode verificar e validar a autenticidade de um dado produto alcançando assim a sua máxima segurança, são os códigos de leitura ótica. Estes códigos podem ser unidimensionais como os códigos de barras ou bidimensionais como o QR-code. Já são vários os estudos que pretendem integrar estes códigos como normas rígidas, de forma a evitar a falsificação das mesmas.

Designando códigos gráficos bidimensionais como uma imagem constituída por pixéis que armazena informações em duas direções, estes fornecem um meio de integrar endereços de páginas da internet, texto genérico, números em formato binário, hexadecimal ou decimal para que sejam possíveis serem lidos por uma câmara. Estes códigos permitem que os utilizadores façam uma leitura apresentando o texto descodificado.

Esta tecnologia foi desenvolvida em 1994 por uma empresa automóvel japonesa, Denso-Wave [1]. Um dos objetivos iniciais seria o rastreio de peças de automóveis de uma maneira mais rápida. Com o desenvolvimento tecnológico, o QR-code é atualmente uma das melhores ferramentas de *marketing*, podendo ser usado em aplicações de realidade aumentada, jogos eletrónicos, para encriptação de dados e

pagamento de compras. Estima-se que até 2025, 2.2 milhares de milhões de utilizadores usem o QR-code como método de pagamento.

Um QR-code pode ser lido em qualquer direção, conseguindo armazenar até 4296 caracteres alfanuméricos [1]. Este usa o algoritmo de Reed-Solomon, um conjunto de algoritmos para correção de erros, para detetar e corrigir potenciais erros que possam haver, logo se tiver algum tipo de dano, consegue-se recuperar até 30% do código danificado [2].

Recentemente, foi desenvolvida uma nova tecnologia anti-contrafação designada por UniQode® [3]. Tal como o QR-code estes códigos de leitura ótica também são bidimensionais, que além de garantirem a identificação do produto e armazenamento dos dados, bastando a utilização de um *smartphone*, acrescentam mais segurança ao método através de várias camadas de segurança.

É formado por várias tecnologias, nomeadamente impressões com *glitter*, hologramas e códigos gráficos, tornando esta tecnologia mais robusta. Assim, em conjunto com um dicionário de símbolos, pertencente só ao fabricante, é possível criar um objeto identificador único. Porém, nesta dissertação só vai ser abordada a tecnologia referente ao código gráfico.

Esta tecnologia é vista como aliada para as autoridades na tentativa do combate à fraude e falsificação de um dado produto, podendo ser aplicada em diversas áreas como por exemplo no ramo da saúde, têxtil ou na indústria alimentar.

## 1.3 Objetivos

Combinando as vantagens do QR-code, como rapidez de leitura na identificação de um produto, alto valor de marketing, elevada utilização, com as vantagens do UniQode®, como a grande capacidade de armazenamento de dados, alto valor estético e que pode ser utilizado através de um *smartphone*, chegamos ao objetivo principal deste trabalho que é a integração dos códigos UniQode® em QR-code. Assim, os vários objetivos ao longo desta dissertação foram:

- Determinação das zonas sujeitas a alteração no QR-code
- Definição e desenvolvimento de criptografia híbrida aplicado ao UniQode®;
- Embutir o UniQode® na zona desejada no QR-code;
- Desenvolvimento de uma aplicação em *iOS* que permite a leitura deste novo modelo.

## 1.4 Contribuições

A implementação deste novo método, que consiste em integrar um código UniQode® num QR-code, contribui para o aumento da segurança na criação e descodificação dos códigos de leitura ótica, de forma a melhorar a validação da autenticidade de uma determinada marca. Isto foi possível através da criação e da definição de uma nova metodologia, em que primeiramente é aplicada uma criptografia híbrida visual à imagem gerada pelo UniQode® e por fim esta é integrada nas zonas passíveis de alteração no QR-code. Além disso, ao integrar esta nova imagem criptografada num código QR-code é possível manter as características do QR-code, que o tornaram numa das maiores ferramentas de marketing junto dos consumidores e numa das maiores tecnologias de identificação de produtos.

Através de uma aplicação desenvolvida para dispositivos *iOS*, a leitura fácil e rápida deste novo código de leitura ótica é assim assegurada.

## 1.5 Estrutura da Dissertação

Este documento será estruturado em 7 capítulos.

No Capítulo 2 será exposta uma revisão da literatura que esteve na base da realização desta dissertação, uma vez que serão apresentados métodos de ocultação de mensagens em códigos gráficos bidimensionais.

De seguida, no Capítulo 3, será abordado o tópico da criptografia onde se destaca a importância da mesma aplicada a sistemas de informação. Serão assim estudados dois métodos criptográficos em que são detalhados os princípios de funcionamento de cada um e explicados os benefícios e limitações de ambos os métodos.

Passando para o Capítulo 4, serão apresentados os códigos de leitura ótica utilizados na realização deste trabalho tais como o QR-code e o UniQode onde será apresentada toda a estrutura e funcionamento destes códigos gráficos bidimensionais. É neste capítulo que serão detalhadas as zonas passíveis a alteração do QR-code que serão fulcrais para a metodologia apresentada nesta dissertação.

Com o intuito de demonstrar um novo método de criptografia híbrida visual foi proposta uma nova metodologia que será exposta e detalhada no Capítulo 5 que consiste em embutir técnicas referentes ao UniQode no QR-code. Com a finalidade de analisar esta metodologia os resultados serão apresentados e discutidos no Capítulo 6.

Por último no Capítulo 7 são abordadas as conclusões dos resultados obtidos e são consideradas algumas sugestões para trabalhos futuros.

# 2

# Revisão de literatura

## Conteúdo

Neste capítulo, são apresentados trabalhos seleccionados, artigos e outras publicações de pesquisa relacionados com o assunto em questão, nomeadamente métodos de ocultação de mensagens em códigos gráficos bidimensionais.

## 2.1 Esteganografia

Esteganografia é a técnica de ocultar informações dentro de outra informação sem haver a possibilidade de distinção. Consiste numa ampla gama de técnicas de comunicação secretas que disfarça a própria existência da mensagem.

A principal fonte de informação são as imagens que nesta nova técnica são usadas como meio de ocultação de outras mensagens. Pode ser realizado através da variação da cor dos pixéis ou outras características, ou a mudança do bit menos significativo (Least Significant Bit (LSB)) do pixel por um bit da mensagem.

O espetro visivel de um ser Humano, que está compreendido entre 370nm (extremo do violeta e 750 nm (extremo do vermelho), não permite distinguir diferenças mínimas de cores. Deste modo, este método pode ser usado em imagens de tons de cinzento ou a cores ao mudar bits para que não seja perceptível.

O método proposto por Cucurull et. al [4], aplica a esteganografia num QR-code. Este método consiste na substituição de blocos de informação do QR-code pela informação a ocultar. Ao ser lido, os blocos de informação substituídos vão ser considerados como erros, e o algoritmo de correção de erros de Reed-Solomon vai tentar corrigi-los, passando assim como um QR-code normal. De maneira a que este método funcione, é importante não alterar os blocos de informação.

Contudo, a maioria dos métodos de esteganografia projetados para imagens digitais não são práticos para imprimir ou digitalizar, pois, irá introduzir uma quantidade significativa de ruído e distorção à imagem, e com isto aos dados ocultos, o que vai significar numa dificuldade, e quase sempre impossibilidade, no bom desempenho dos esquemas de correção de erros.

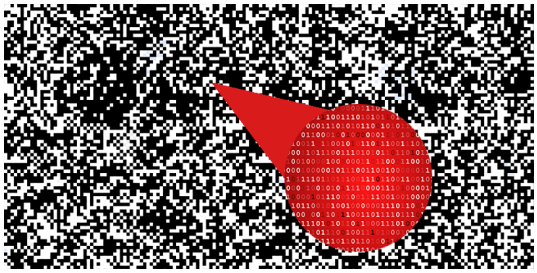


Figura 2.1: Exemplo de esteganografia.

## 2.2 Marca d'água

A marca d'água digital é uma técnica de proteção de direitos autorais (copyright) para multimédia. Esta técnica também pode ser usada para verificar a autenticidade de documentos digitais.

A marca d'água digital usada em imagens é desenvolvida com o objetivo de ter robustez, imperceptibilidade e segurança da imagem original. Deve ser capaz de resistir à remoção da marca d'água e a sua utilização não deve ser detetada a partir de análises à imagem. Porém já existem formas de remover a marca d'água a imagens.

Figura 2.2: Exemplo de *watermarking*.

## 2.3 Criptografia Visual

O conceito de criptografia visual foi implementado inicialmente por Naor e Shamir em 1994 [5]. Criptografia visual é outro tipo de esteganografia que usa as características da visão humana para desencriptar uma imagem se a imagem-chave for usada corretamente. Este conceito encripta uma imagem dividindo-a em múltiplas imagens, permitindo que a imagem secreta seja recuperada quando várias imagens-chave são sobrepostas.

Na criptografia visual não é possível obter nenhuma informação perceptível a partir de cada imagem-chave. Só depois de serem sobrepostas, é que é possível obter a imagem secreta, mas na imagem secreta não é possível obter nenhuma informação das imagens-chave (Fig. 2.3).


Figura 2.3: (a) QR-code original; (b) Imagem secreta; (c) Imagem-chave 1; (d) Imagem-chave 2; (e) Simulação das imagens-chave sobrepostas [6].

A forma mais fácil de implementar criptografia visual é imprimir as duas imagens-chaves num papel transparente. A sobreposição destas duas imagens, juntamente com uma fonte de luz cria o equivalente a uma operação matemática lógica *OR*.

Comparado com a criptografia digital, em que são implementadas com algoritmos computacionais, a criptografia visual pode ser categorizada como uma técnica de criptografia ótica [6].

O uso de QR-code também tem sido usado na criptografia visual [6, 7], no qual o QR-code é dividido em duas imagens-chave. O QR-code original é recuperado ao sobrepor as duas imagens-chave, e de seguida a informação é obtida ao ler o QR-code.

## 2.4 Dithering

*Dithering* é usado regularmente para imprimir imagens monocromáticas. Um exemplo são as fotografias dos jornais. É possível apresentar uma imagem com vários tons de cinza numa imagem com apenas dois tons, usando esta técnica. Consiste em mapear a imagem em níveis de cinza numa imagem binária, pixéis pretos e brancos, em que a densidade local dos pixéis pretos e brancos faz com que se aproximem do nível médio de cinza da imagem original (Fig. 2.4). *Dithering* é normalmente usado para apresentar imagens monocromáticas ou com dois tons [8–10].

Figura 2.4: Exemplo de *Dithering*.

O método usado para a criação e descodificação de códigos gráficos do UniQode [8] é baseado em *dithering* de forma a criar imagens em meio-tom (*Halftone*), alterando determinados pixéis de modo a ocultar informação.

# 3

# Criptografia

## Conteúdo

A arte de escrever e resolver códigos com o objetivo de disfarçar dados é denominada de criptografia. O princípio básico da criptografia é transformar dados numa forma ininteligível para um humano. Trata-se de encontrar e aperfeiçoar métodos de envio de mensagens entre duas entidades sem risco de terceiros não autorizados acederem ou entenderem as mensagens enviadas. As suas origens datam de vários séculos atrás, quando foi usado por Julius Caesar para fins militares, onde era usada uma simples cifra de substituição. A sua função era substituir uma letra do alfabeto pela letra que se encontra três posições atrás (fig. 3.1).

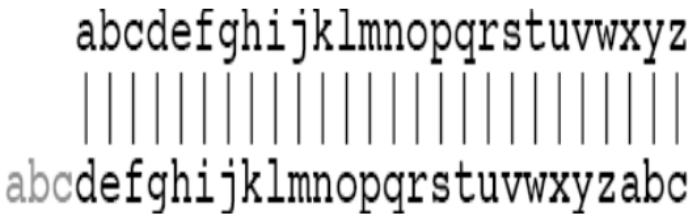



Figura 3.1: Cifra de Julius Caesar - substituição simples [11].

As técnicas de criptografia de hoje são muito mais complexas e avançadas. São implementados algoritmos complexos para converter informações num formato incompreensível.

Criptografia também se refere à arte de encriptação de uma mensagem, ou seja, escrever uma mensagem numa forma incompreensível para quem não conhece o processo de encriptação. Uma mensagem encriptada é chamada de texto cifrado, enquanto que uma mensagem que pode ser compreendida pelo seu remetente e destinatário, mas também por qualquer pessoa que possa obtê-la, é chamada de texto claro [12]. O processo que consiste em restaurar o texto cifrado num texto claro é a desencriptação. Uma chave é usada para encriptar ou desencriptar dados. Pelo princípio de *Kerckhoffs* a segurança do sistema criptográfico deve residir apenas num único parâmetro, a chave. Ou seja, o sistema deve ser seguro se tudo sobre o sistema é conhecido publicamente, exceto a chave [13].

Hoje, o objetivo da criptografia não se foca só no problema da comunicação, mas também com qualquer problema relacionado com computação, relativamente a ataques internos ou externos.

A criptografia é atualmente um tópico importante na ciência da computação. Algumas metas de segurança devem ser alcançadas para tornar qualquer sistema criptográfico seguro. Assim, a criptografia moderna tem como base as seguintes características:

- Garantir a confidencialidade de informação entre duas entidades, sendo in-

compreensível aos restantes.

- Autenticação para que o remetente seja corretamente identificado.
- Garantir a integridade da mensagem, para que não haja alteração da informação por pessoas não autorizadas.
- Impedir que o remetente de uma mensagem negue o envio de uma mensagem, ou não-repúdio.
- Garantir a disponibilidade dos dados sempre que necessário.

Qualquer tentativa em atacar as características da criptografia resulta num problema à segurança e integridade do sistema.

A criptografia passou por uma revolução no final do século XX com o aparecimento de computadores e outros dispositivos eletrónicos. Essas tecnologias tornaram mais fácil criar cifras mais complexas e também mais facilidade nos ataques. A criptografia moderna é concebida de maneira a que haja uma computação elevada, como a fatorização de inteiros ou de logaritmos discretos. Estes são problemas fáceis de definir, mas difíceis de resolver. São considerados computacionalmente seguros, mesmo que teoricamente sejam possíveis de ser quebrados, mas praticamente impossível pelo tempo que demora os ataques.

O uso de criptografia é necessário se a proteção e privacidade dos dados for crucial. Existem dois tipos principais de criptografia: criptografia simétrica e criptografia assimétrica.

## 3.1 Criptografia simétrica

A criptografia simétrica, também conhecida por criptografia de chave secreta, depende da existência de uma chave secreta partilhada por ambas as entidades antes de qualquer partilha de mensagens. O remetente utiliza a chave secreta para a encriptação da mensagem, transformando-a em texto cifrado, incompreensível para qualquer outra pessoa. O destinatário usa a mesma chave para decifrar a mensagem de volta à mensagem original, ou texto simples. Na figura 3.2 pode-se ver o processo da criptografia simétrica.

A principal desvantagem da criptografia simétrica é que todos os envolvidos precisam trocar a chave privada antes de haver qualquer troca de mensagens.

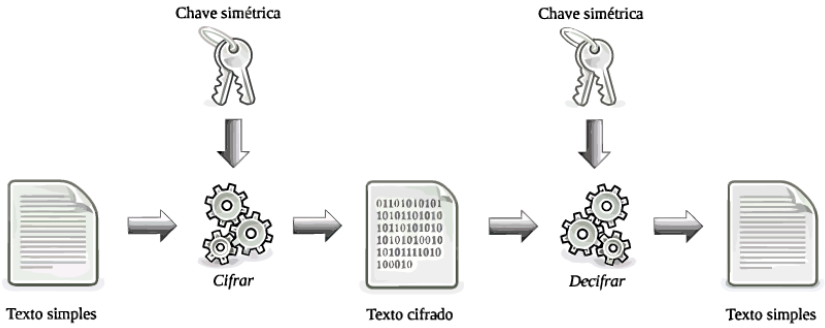



Figura 3.2: Processo da criptografia simétrica [14].

### 3.1.1 Advanced Encryption Standard (AES)

AES, ou Padrão de Criptografia Avançado, é o esquema de criptografia simétrica recomendado pelo Instituto Nacional de Padrões e Tecnologia (NIST) dos EUA [15]. Veio substituir o seu antecessor que começou a mostrar-se insuficientemente seguro, o Data Encryption Standard (DES), depois do NIST ter organizado uma competição para descobrir um algoritmo superior. Era necessário um novo método de criptografia simétrica porque o DES usava uma chave de tamanho relativamente pequena, de 56 *bits*, e era possivelmente vulnerável a ataques de força bruta. Tinha um número máximo de combinações possíveis de $2^{56} = 7.2 \cdot 10^{16}$, que comparado com o número máximo para uma chave de 256 *bits* utilizada pelo AES é de $2^{256} = 1.1 \cdot 10^{77}$ combinações.

É baseado no algoritmo de Rijndael [16], criado por Joan Daemen e Vincent Rijmen em 1999, sendo atualmente o padrão para a criptografia simétrica.

AES é uma cifra de chaves simétricas por blocos com um tamanho de 128 *bits* e com chaves de 128, 192 e 256 *bits*. Cada bloco de 128 *bits* é agrupado de forma a produzir uma matriz 4x4, com cada posição a ter 8 *bits* (Fig. 3.3).

| | | | |
|---|---|---|---|
| $b_{00}$ | $b_{01}$ | $b_{02}$ | $b_{03}$ |
| $b_{10}$ | $b_{11}$ | $b_{12}$ | $b_{13}$ |
| $b_{20}$ | $b_{21}$ | $b_{22}$ | $b_{23}$ |
| $b_{30}$ | $b_{31}$ | $b_{32}$ | $b_{33}$ |

Figura 3.3: Estrutura de um bloco de 128 *bits*.

É baseado numa rede de substituição-permutação, isto é, uma cifra produto composta por um número de iterações, cada uma envolvendo substituições e permutações [17]. O número de iterações é o seguinte:

- 10 iterações para chaves de 128 *bits*;
- 12 iterações para chaves de 192 *bits*;
- 14 iterações para chaves de 256 *bits*.

Cada iteração consiste nas seguintes operações:

1. Esta operação é realizada antes de qualquer iteração:
    - AddRoundKey - Cada bloco é combinado com uma chave, derivada da chave principal, usando transformações lógicas XOR.
2. Para cada uma das 9, 11 e 13 iterações intermédias são aplicadas as seguintes transformações:
    - SubBytes - Cada byte do bloco é substituído pelo seu valor numa *lookup-table* de 8 bits.
    - ShiftRows - Os bytes em cada linha são deslocados para a esquerda de acordo com o número de linha (começando em zero).
    - MixColumns - Cada coluna é transformada multiplicando por uma matriz fixa.
    - AddRoundKey
3. Finalmente para a última iteração (10, 12 e 14):
    - SubBytes
    - ShiftRows
    - AddRoundKey

Geralmente, especialistas em segurança consideram o AES seguro contra ataques de força bruta, nas quais todas as combinações possíveis de chaves são verificadas até que a chave correta seja encontrada. No entanto, o tamanho da chave influencia o tempo de duração destes ataques. Quanto maior a chave, menos hipóteses um computador moderno tem de decifrar o conteúdo da mensagem. Assim, uma chave de 256 *bits* é significativamente melhor do que uma chave de 128 *bits*.

Todavia, nem sempre usar uma chave de 256 *bits* é o ideal. Quando se pensa em utilizar um dispositivo mais antigo ou em que o tempo de resposta é importante, esta chave pode não ser a melhor opção, pois requer também mais poder de processamento para ser gerada. Neste caso, pode ser mais adequado usar uma chave de 128 *bits*.

## 3.2 Criptografia assimétrica

Criptografia assimétrica, também conhecida por criptografia de chave pública, foi criado por Whitfield Diffie e Martin Hellman em 1976, como uma solução para o problema das trocas de chaves. Ao contrário do que acontece na criptografia simétrica, aquela usa duas chaves: uma chave pública que é usada para cifrar e uma chave privada usada para decifrar a mensagem. Assim, em vez de haver uma chave para encriptar e desencriptar, cada participante tem a sua própria chave, uma chave pública e uma privada.

Na criptografia assimétrica, a chave pública é conhecida publicamente para que qualquer pessoa possa usá-la para encriptar a mensagem. A chave privada só é conhecida pelo seu criador e é usada para decifrar a mensagem. Na Figura 3.4 pode ver-se o processo da criptografia assimétrica.

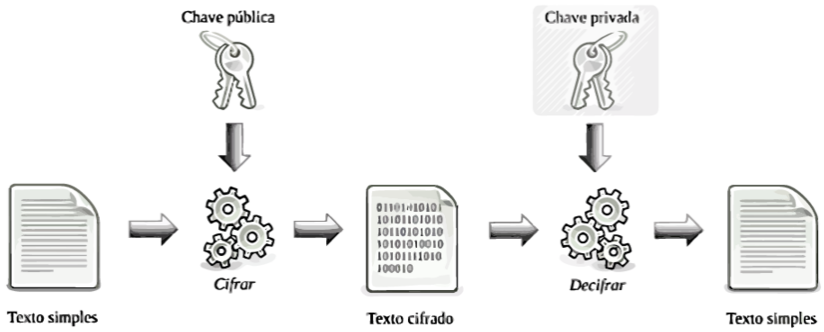



Figura 3.4: Processo da criptografia assimétrica [14].

Para que duas entidades comuniquem de um modo seguro, cada uma tem de ter a chave pública do outro, que por definição está disponível ao público, mantendo as suas chaves privadas em segurança. O remetente encripta a mensagem com a chave pública do destinatário e este usa a sua chave privada para desencriptar.

Comparativamente à criptografia simétrica, a principal desvantagem da criptografia assimétrica é que esta é mais demorada. Isto ocorre por causa da complexidade matemática e, portanto, requer muito mais poder computacional. Enquanto que a principal vantagem é a de não haver necessidade de trocar de chaves.

Qualquer sistema que use criptografia assimétrica, com um par de chaves pública/privada, deve ser:

- Fácil criar o par de chaves.
- Fácil encriptar a mensagem com a chave pública.
- Fácil desencriptar a mensagem usando a chave privada.
- Difícil descobrir a chave privada com base na chave pública.

O desenvolvimento de métodos com criptografia de chaves assimétricas, permitiu resolver o problema da distribuição das chaves. Combinada com a confidencialidade oferecida pela encriptação, a criptografia assimétrica garante autenticação, não-repúdio assim como a integridade da mensagem.

### 3.2.1 Rivest–Shamir–Adleman (RSA)

O algoritmo de RSA, foi o primeiro sistema revolucionário de criptografia assimétrica a ser publicado em 1977 [18] baseado na proposta de Diffie-Hellman por Ron Rivest, Adi Shamir e Leonard Adleman. A sua implementação resulta na escolha de dois números primos relativamente grandes, que depois são multiplicados. A segurança do algoritmo de RSA vem da complexidade da factorização do número resultante desta multiplicação. Esta dificuldade é o que torna o RSA competente. É fácil calcular o produto de dois números inteiros, mas a operação inversa já é impraticável e demorada. Desta maneira, o RSA é seguro enquanto a descoberta destes dois números for inexequível e impossível num tempo útil. Por este motivo, é aconselhado escolher chaves com um tamanho relativamente grande, já que esta influencia o nível de segurança. Normalmente são usadas chaves de 2048 bits ou superiores.

RSA é um dos algoritmos mais utilizados e considerado para métodos de criptografia assimétrica. No entanto, devido à grande computação requerida relativamente à criptografia simétrica, é usado um sistema híbrido [19] para quando se pretende encriptar ficheiros ou informação de maior dimensão. Este sistema é formado pelo AES e pelo RSA e consiste no processo de encriptação pelo algortimo de AES, enquanto que o RSA na sua vez, vai encriptar a chave utilizada pelo algoritmo de AES. Esta solução híbrida resolve os problemas de desempenho e da segurança de troca de chaves, constituindo assim uma vantagem.

O algoritmo de criação de chaves de RSA consiste nas seguintes etapas:

- Escolha de dois números primos aleatórios, **p** e **q**, grandes em magnitude mas diferentes em comprimento por alguns dígitos.
- Multiplicar estes dois números obtendo o módulo $n = p \cdot q$.
- Obter o tociente de Euler, $\phi(n) = mmc(p-1, q-1)$, em que *mmc* é o mínimo múltiplo comum entre os dois números.
- Finalmente escolher um número inteiro $e$ de tal forma que satisfaça as seguintes condições:
  - $1 < e < \phi(n)$
  - $e$ e $\phi(n)$ sejam primos entre si
  - $mdc(e, \phi(n)) = 1$, *mdc* denota o máximo divisor comum.

A chave pública consiste no número inteiro $e$ e módulo $n$. A chave privada consiste no número inteiro $d$.

$$d = e^{-1} mod(n)$$

Os números inteiros $p$, $q$ e $\phi(n)$ não pertencem ao par de chaves. O processo de encriptação resume-se à computação do texto cifrado, $c$, a partir da equação 3.1.

$$c = m^e mod(n) \tag{3.1}$$

A desencriptação é o processo inverso (Eq. 3.2), obtendo assim a mensagem original, $m$.

$$m = e^d mod(n) \tag{3.2}$$

# 4

# Códigos de Leitura Ótica

## Conteúdo

Códigos de leitura ótica, ou *Machine Readable Codes (MRC)*, foram inventados com a intenção de aumentar a rastreabilidade, identificação rápida, fácil e segura de produtos [20]. Um dos primeiros *MRC*, e ainda muito usado, foi o código de barras (Figura 4.1). O código de barras é um código gráfico uni-dimensional que armazena várias informações sobre o produto numa só direção. Os padrões para este código são geridos pela GS1 [20], fornecendo uma linguagem comum que garante a eficiência dos processos de negócio e proporcionando formas de reduzir custos por meio da automação baseada numa identificação global única.


Figura 4.1: Exemplo de um código de barras do tipo GS1-128.

Porém, com o avanço da tecnologia, uma nova técnica veio aumentar a capacidade máxima dos códigos de barras, que atualmente é de 100 caracteres, nomeadamente chamada de códigos gráficos bidimensionais.

## 4.1 QR-code

*QR-code* é um código gráfico bidimensional, ou seja, os pixéis estão agrupados numa matriz bidimensional. São considerados como a evolução dos códigos de barras por conseguirem armazenar mais informação, e consequentemente requer também um programa mais sofisticado para a sua descodificação.

O QR-code foi desenvolvido e criado pela empresa japonesa Denso Wave em 1994. O seu ponto forte é a rapidez de armazenar informação e a rapidez de leitura. Atualmente, o QR-code consegue ser lido pela maioria dos *smartphones* o que faz com que seja mais um meio de comunicação por parte das empresas.

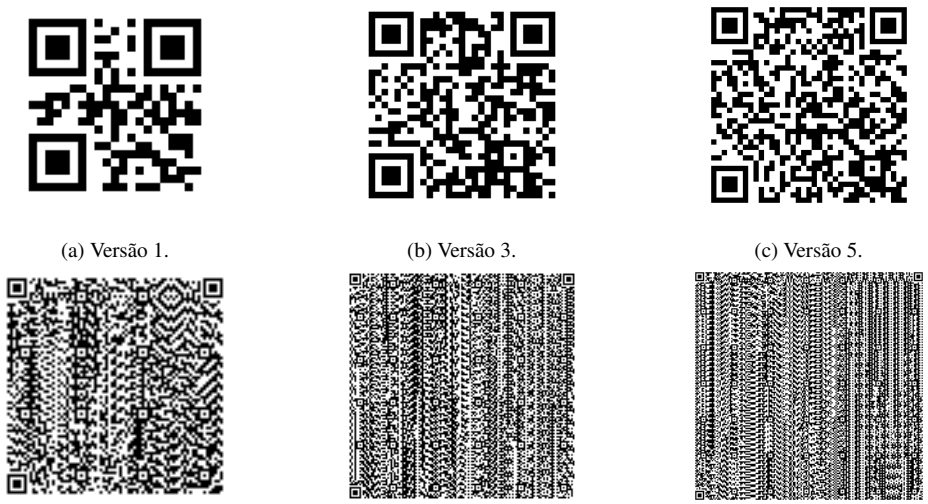
Existem 40 versões do QR-code, variando da versão 1 à versão 40, cada uma é constituída por módulos pretos e brancos. Cada versão tem uma dimensão e um padrão específico. Isto significa que cada versão tem um determinado número de módulos, assim como um limite total de quantidade de dados. A primeira versão é constituída por 21x21 módulos, e com o aumentar das versões é aumentada em 4 módulos adicionais comparado com a versão anterior, até à versão 40, que é constituída por 177x177 módulos. O número de módulos pode ser descrito pela equação 4.1. Contudo, a versão a ser usada depende da quantidade de informação e do tipo de informação (alfanumérica, binária, Kanji [1] ou numérico) a serem codificados,

[1] Kanji são caracteres da língua japonesa adquiridos a partir de caracteres chineses.

bem como o nível de correção de erros. Além disso, o QR-code consegue guardar até 7089 caracteres.

$$N_{módulos} = 4 \cdot Versão + 17 \tag{4.1}$$

Na Figura 4.2 pode-se ver alguns exemplos de QR-code:



(a) Versão 1. (b) Versão 3. (c) Versão 5.

(d) Versão 15. (e) Versão 30. (f) Versão 40.

Figura 4.2: Algumas versões do *QR-code*.

### 4.1.1 Estrutura de um QR-code

Um QR-code consiste em várias regiões de codificação e de padrões de funcionamento essenciais à sua descodificação e deteção. Um exemplo destas regiões é ilustrado na figura 4.3.

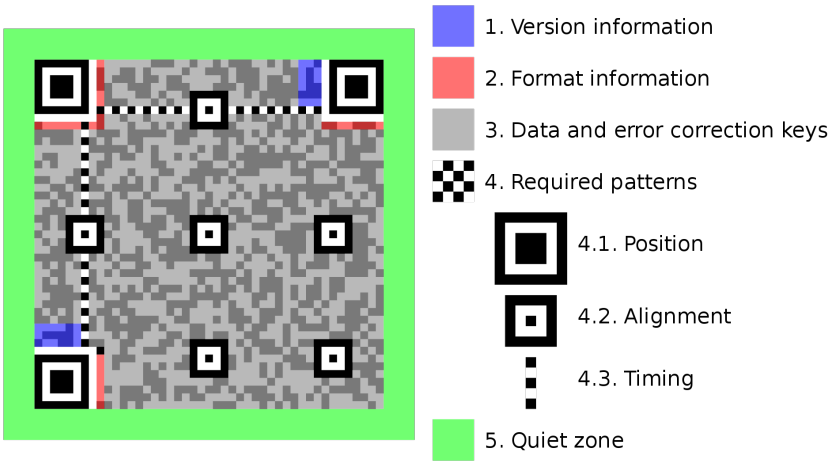



Figura 4.3: Estrutura de um *QR-code*, destacando os elementos funcionais.

Os padrões de funcionamento não fazem parte da codificação da informação no QR-code, estes consistem nos padrões de tempo, de alinhamento e de localização:

• **Padrões de tempo**: Sequência de módulos pretos e brancos alternados que ajuda o programa de descodificação/localização a determinar o tamanho de cada módulo em pixéis.

• **Padrões de alinhamento**: Ajuda ao programa corrigir imagens com possíveis distorções. Cada versão consiste num número diferente de padrões de alinhamento. Apenas na versão 1 do QR-code não é apresentado este padrão.

• **Padrões de localização**: Estes padrões consistem nas 3 estruturas idênticas localizadas nos cantos superiores e no canto inferior esquerdo. Permite ao programa que identifique o QR-code e determine a orientação correta. É composto por um quadrado preto 7x7 com outro quadrado branco 5x5 e por fim preenchido por módulos pretos.

### 4.1.2 Região de codificação e geração do símbolo QR-code

A capacidade do QR-code depende de vários fatores: do tipo de informação (alfanumérica, binária, Kanji ou numérico), da versão e do nível do algoritmo de correção de erros. Quanto maior o nível do algoritmo de correção de erros, menos blocos de informação consegue-se colocar, pois este vai necessitar de mais blocos para correção de erros. Mas existe mais possibilidades de ser descodificado, no caso, se o QR-code estiver danificado ou sujo. Esta característica permite também que se possa incorporar uma fotografia (Fig. 4.4) ou outras informações dentro de um QR-code. Uma mensagem é dividida num fluxo de bits e cada bloco é formado por 8 bits, que no símbolo final é representado pelos módulos pretos e brancos.

A codificação do algoritmo de correção de erros fornece condições essenciais à extração do fluxo de bits correto e descodificação da mensagem do QR-code em cenários ruidosos. Isto é, se a imagem estiver corrompida ou desfocada, podemos perder informações por não detectar corretamente os bits.

Figura 4.4: Imagem incluída no QR-code.

O algoritmo de criação do QR-code implementa o código de Reed-Solomon para deteção e correção de erros aos dados originais. Existem quatro níveis para correção de erros (Tabela 4.1). Assim aumentar este nível aumenta também a habilidade de corrigir uma maior quantidade de erros. Este nível é escolhido consoante o ambiente em que vai ser usado.

| **Nível de correção de erros** | **Capacidade de correção de erros** |
|---|---|
| *L (Low)* | 7% |
| *M (Medium)* | 15% |
| *Q (Quartile)* | 25% |
| *H (High)* | 30% |

Tabela 4.1: Nível de correção de erros e correspondente percentagem de dados que se pode restaurar

Se uma mensagem tiver de ser codificada com um nível de correção de erros superior, poderá ser necessária uma versão do QR-code maior. Deste modo, a capacidade máxima de armazenamento ocorre para uma versão 40 com o nível de correção de erros baixa (Low). É por este motivo que é, no caso geral, preferido o nível baixo de correção de erros, pois permite maior capacidade de informação.

Os blocos de informação são constituídos pela mensagem e pelo código de correção de erros. São codificados de maneira a que os blocos da mensagem e os blocos da correção de erros se intercalem. Isto é realizado para reduzir a possibilidade de que danos façam com que o QR-code se torne indecifrável. A Figura 4.5 mostra a organização dos dois tipos blocos de informação para a versão 4.

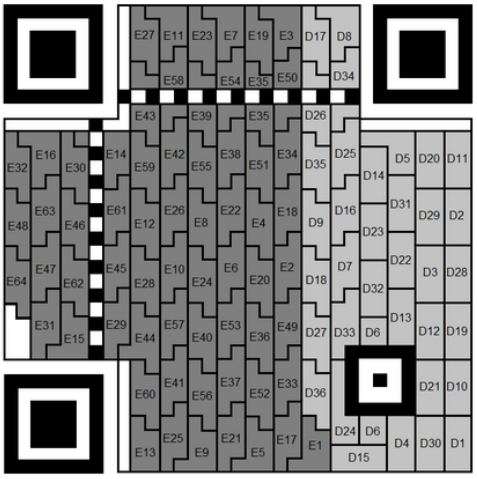



Figura 4.5: Blocos de informação de um QR-code [2], em que os blocos **D** correspondem a informação e **E** a correção de erros.

Depois de codificar os blocos de informação, é aplicada uma máscara, realizando uma operação binária XOR (Fig 4.6) aos blocos de informação, os outros padrões não são afetados por esta operação. Esta é escolhida automaticamente pelo programa enquanto cria a matriz final. O objetivo da máscara é uma distribuição equilibrada entre os módulos pretos e brancos.

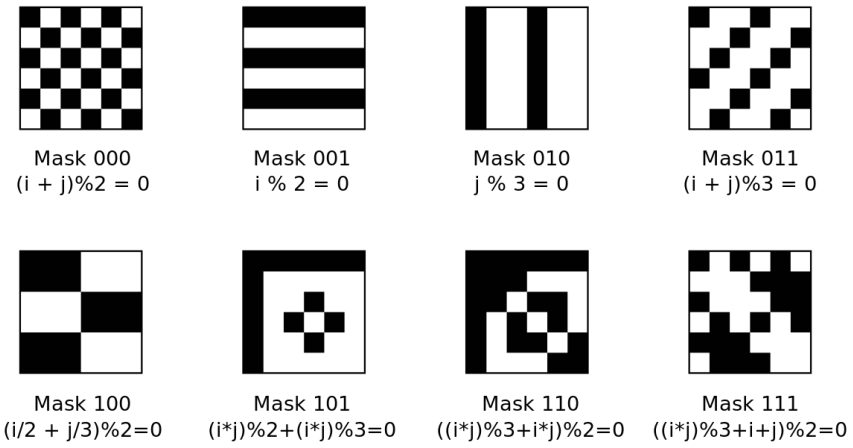



Figura 4.6: Máscaras usadas num QR-code.

## 4.2 UniQode

Um grupo de investigadores do ISR da Universidade de Coimbra em parceria com a INCM introduziram um novo *MRC*, designadamente o código UniQode Este é constituído por várias camadas de segurança, garantindo a autenticidade de um produto e permitindo também rapidez na leitura e na validação da autenticidade da marca. Como o *QR-code*, o UniQode pode também ser usado para a identificação de produtos e preços. O UniQode é um sistema de criptografia simétrica que usa a mesma chave para cifrar e decifrar a mensagem. Este é constituído por um conjunto de tecnologias complementares, designadamente impressões com glitter, hologramas e códigos gráficos. A tecnologia de impressão com glitter permite criar um identificador único por objeto, tornando este tipo de tecnologia, em termos de segurança, muito difícil de falsificar. Porém para o efeito desta dissertação, foi só considerado o código gráfico do UniQode.

A aplicação para este tipo de código permite que seja codificado em imagens baseadas em pixéis ou em ícones [3].

### 4.2.1 Dicionário

A criação do UniQode consiste na encriptação de mensagens utilizando um dicionário próprio (Fig. 4.7). Este dicionário é constituído com base em pixéis organizados em células que podem ser formadas por diversas formas. A célula mais comum é constituída por 3x3 pixéis.

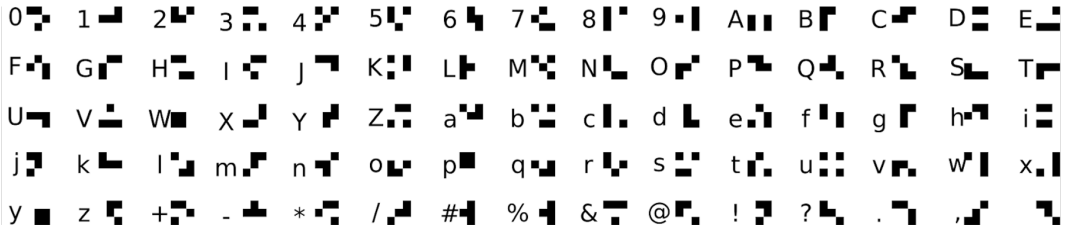


Figura 4.7: Exemplo de um dicionário do UniQode [3].

Existem várias distribuições possíveis que podem ser agrupadas para a formação deste dicionário de acordo com o número de pixéis pretos ou brancos que existem numa célula. Cada uma destas distribuições é chamada de *quantum*. Para uma célula de 3 por 3 pixéis existem 10 *quanta* possíveis (Fig. 4.8).

| Quantum | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Distribution Black/White | 9/0 | 8/1 | 7/2 | 6/3 | 5/4 | 4/5 | 3/6 | 2/7 | 1/8 | 0/9 |
| Quantity of Patterns | 1 | 9 | 36 | 84 | 126 | 126 | 84 | 36 | 9 | 1 |
| Gray Scale Range | 0-26 | 27-51 | 52-76 | 77-102 | 103-127 | 128-153 | 154-178 | 179-204 | 205-229 | 230-255 |
| Quantized Color | | | | | | | | | | |
| Example of Pattern | | | | | | | | | | |

Figura 4.8: Diferentes distribuições de *quanta* [3].

Como se pode ver pela Figura 4.8 existem várias distribuições para uma célula de 3x3 pixéis e com isto diferentes quantidades de padrões para os diferentes *quanta*. Isto quer dizer que se pode utilizar inúmeros dicionários dependendo da utilização pretendida. Pois cada letra, número ou símbolo pode ter um padrão de pixéis pretos e branco diferentes para o mesmo *quantum*, isto é, numa imagem podem ser usados vários dicionários em que a mesma letra pode ter diferentes distribuições de pixéis. Isto pode ser útil pois determinados utilizadores podem só ter acesso a certos dicionários.

### 4.2.2 Codificação

O método utilizado no UniQode usa a técnica de *dithering* para criar imagens de meio-tom (Halftone), denominado de código gráfico. Dividindo cada pixel da imagem em k x k pixéis, normalmente 3x3 pixéis, dependendo do dicionário a usar, reduzindo a tonalidade de cinzento para preto ou branco, mas mantendo uma distribuição de pixéis pretos e brancos de forma a manter a aparência dos níveis de cinzento, e ao mesmo tempo aumentando a resolução da imagem (Fig. 4.9). Podendo também ser gerado simplesmente uma matriz de pixéis pretos e brancos.

O código gráfico do UniQode determina na imagem original células candidatas de acordo com o *quantum* utilizada no dicionário. Estas células são substituídas pelo padrão correspondente à mensagem e de acordo com o tamanho. As células que não foram utilizadas para a codificação da mensagem e as que não foram seleccionadas como candidatas são substituídas com padrões não pertencentes ao dicionário. De referir que numa imagem podem ser codificadas várias mensagens utilizando diferentes *quanta* e com isso usar diferentes dicionários. Os *quanta* utilizados são definidos por pessoas autorizadas.

Figura 4.9: Exemplo do processo de codificação usando o método do UniQode cedido por VISTeam.

### 4.2.3 Descodificação

Para a descodificação, o UniQode vai percorrer a imagem gerada e comparar com o dicionário. Se houver alguma combinação de pixéis que não correspondem ao que existe no dicionário, este conjunto é descartado como não pertencente à mensagem. Isto faz com que seja possível ter uma imagem com uma maior quantidade de células do que a dimensão da mensagem a encriptar.

Uma característica importante do UniQode é que este não é limitado como no QR-code, em que é restrito por uma versão máxima. A versão 40 do QR-code consegue codificar até 4296 caracteres no modo alfanumérico, mas apesar do UniQode só conseguir 3481 caracteres para o mesmo tamanho de 177x177, este não tem limite de tamanho [8].

### 4.2.4 Aplicações da tecnologia UniQode

Atualmente a tecnologia UniQode em conjunto com as autoridades de regulamentação de produtos têm cooperado para a mitigação do problema da contrafação, garantindo autenticidade de uma marca e segurança aos consumidores.

A primeira aplicação desta tecnologia foi na indústria tabaqueira, mais concretamente em selos fiscais dos maços de tabaco. Estão já em implementação desde Maio de 2019, separando os malefícios relacionados com a saúde, traz a garantia aos consumidores que aquele produto foi verificado e validado corretamente, o que proporciona maiores retornos financeiros à indústria e a devida arrecadação de impostos pelo Estado.

Esta tecnologia já está aplicada ao Documento Único Automóvel (DUA) desde setembro de 2020 e será ainda aplicada a garrafas de vinho. No entanto esta pode vir a ser alargada a diversas indústrias como a indústria farmacêutica, a alimentar, a têxtil e calçado ou podendo ser usado em marketing e comunicação. Alguns dos exemplos de selos fiscais que usam a tecnologia do UniQode podem ser observados nas Figuras 4.10 e 4.11.


Figura 4.10: UniQode aplicado a selos fiscais nos maços de tabaco.

Figura 4.11: UniQode aplicado a selos de proteção de marca em garrafas de vinho.

# 5

# Metodologia

## Conteúdo

Nesta dissertação pretendeu-se desenvolver um novo método de encriptação. Este novo método consiste em implementar as técnicas anteriormente referidas ao UniQode e posteriormente embutir este no QR-code, garantido assim uma maior robustez e confidencialidade da informação. Neste capitulo irá ser detalhada a metodologia usada para a obtenção deste novo modelo criptográfico.

## 5.1 Algoritmo

Para a implementação do método anteriormente descrito, foi proposto nesta dissertação um algoritmo. Este algoritmo tem de ter a capacidade de encriptar a imagem gerada pelo UniQode. O QR-code terá que se adaptar de forma a que seja possível embutir o UniQode no mesmo. A propriedade detalhada anteriormente em 4.1.2, nomeadamente a região de codificação do QR-code, foi levada em conta na resolução deste problema. Deste modo, o esquema da Figura 5.1 representa as diversas fases do processo para que todas as condições se verifiquem.

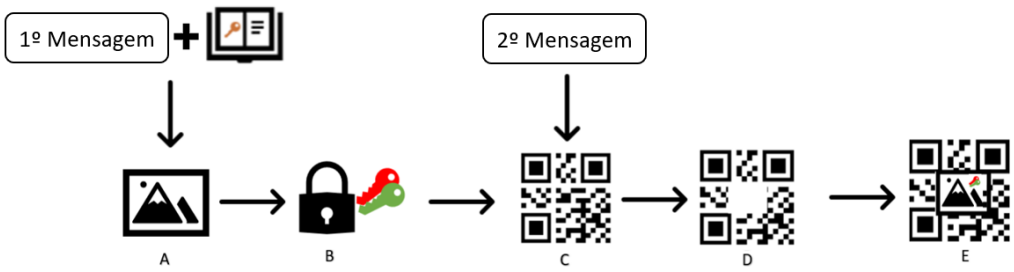



Figura 5.1: Esquema do Algoritmo Implementado.

Relativamente ao esquema da Figura 5.1, destacam-se quatro fases:

• Fase A - onde foi gerada uma imagem através do UniQode, criptografada simetricamente através do dicionário de símbolos (4.2);

• Fase B - onde foi aplicada criptografia à imagem gerada pelo UniQode;

• Fase C - onde foi gerado um dado QR-code com uma mensagem diferente do UniQode;

• Fase D - onde foi identificada a área sujeita a alterações do QR-code gerado em C (4.1.2 ; 5.3);

• Fase E - por último, onde foi embutido o UniQode no QR-code.

Todas estas fases serão detalhadas nas seguintes subseções.

### 5.1.1 Ferramenta de desenvolvimento

A escolha das respetivas ferramentas de desenvolvimento para o algoritmo estão relacionadas com a vasta documentação e bibliotecas tanto da linguagem *C++*, como do *Python*.

Deste modo, é aplicado as vantagens de ambas as linguagens ao nosso caso, onde no C++ existe uma biblioteca própria do UniQode e de outra biblioteca de manipulação de imagens denominada *OpenCV*. Esta biblioteca também está disponível em *Python*. Além disso, o *Python* possui também uma biblioteca, designada por *Pycryptodme*, que será utilizada como auxiliar na respetiva criptografia. Esta biblioteca, oferece assim todas as funções de segurança necessárias à implementação do método nesta dissertação.

## 5.2 Criptografia Híbrida aplicada ao UniQode

A criptografia de imagens é um dos meios mais eficazes para garantir a segurança de imagens transmitidas pela internet. Um dos métodos usados é utilizar sistemas caóticos [21, 22]. Porém, a imagem criptografada usada nestes métodos gera pixéis RGB ou pixéis com tons de cinza. Todavia não é o que se pretende, pois o que se deseja é que o código gráfico não mude de cor, isto é, que mantenha apenas pixéis pretos e brancos, no caso dos códigos de leitura ótica.

As vantagens de implementar um código gráfico embutido noutro é que na imagem final não se nota distinção para um QR-code normal.

### 5.2.1 Contexto

O UniQode faz uso da criptografia simétrica, através de um dicionário próprio de símbolos, em que este faz a função da chave privada. Porém as limitações apresentadas no Capítulo 4 referentes à partilha de chaves na criptografia simétrica mostra que isto poderá ser um problema. Seria problemático se fosse possível ter acesso à chave privada, e assim ter a possibilidade de descodificar o conteúdo da mensagem ou, pior ainda, poder gerar novos códigos válidos com uma mensagem escolhida por falsificadores.

Nesta dissertação foi utilizada uma combinação da criptografia simétrica com a criptografia assimétrica. Esta solução, chamada de criptografia híbrida combina a conveniência da criptografia de chave pública com a eficiência da criptografia de chave privada.

### 5.2.2 *Padding*

Na prática, usar o algoritmo de RSA só em si não é seguro, como já foi provado em vários ataques [23, 24]. O algoritmo original do RSA é determinístico, isto é, para o mesmo par de chaves e o mesmo texto claro, vai gerar sempre o mesmo texto cifrado. Não existe nenhuma variável aleatória. Esta propriedade torna o algoritmo vulnerável a um ataque de texto claro cifrado, onde é possível obter textos cifrados aleatórios de textos claros conhecidos e comparar com textos cifrados obtidos previamente. Com esta comparação, é possível obter o texto original sem existir qualquer processo de desencriptação.

Para contornar este problema existem métodos que implementam variáveis aleatórias no processo de encriptação. Para obter a segurança necessária ao sistema, deve-se aplicar um pré-processamento ao texto claro antes de qualquer operação de criptografia. O pré-processamento consiste em adicionar um preenchimento de bits aleatórios ao texto claro até obter a mesma dimensão que o módulo de RSA. Isto é realizado pelo *Optimal Asymmetric Encryption Padding (OAEP)*, criado por Bellare e Rogaway em 1994 [25]. OAEP usa um gerador de números pseudo-aleatórios para assegurar que o texto cifrado seja sempre diferente. Contudo, isto influencia no tamanho dos dados a encriptar, pois

OAEP satisfaz os seguintes objetivos:

• Adiciona um elemento aleatório que pode ser usado para converter um sistema determinístico num sistema probabilístico.

• Impossibilita a desencriptação parcial de textos cifrados, assegurando que um competidor não possa recuperar nenhuma parte do texto claro.

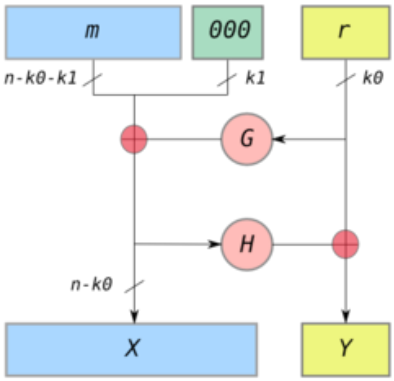



Figura 5.2: Exemplo de um preenchimento, em que n é o número de *bits* no módulo de RSA, $k_0$ e $k_1$ são números inteiros determinados pelo protocolo, G e H são funções hash criptográficas e $\oplus$ uma operação matemática lógica XOR.

### 5.2.3 Implementação

As etapas para a implementação desta fase são indicadas no esquema da Figura 5.3.

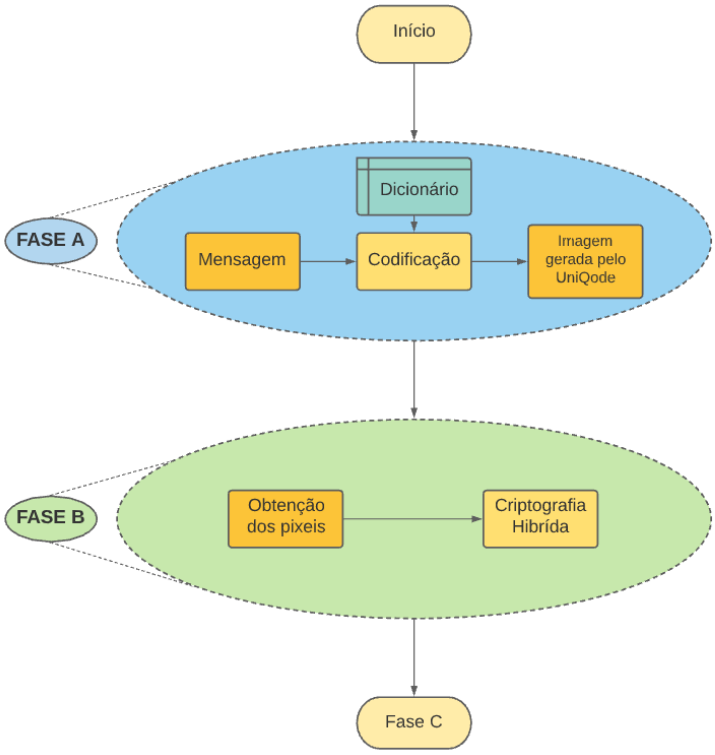



Figura 5.3: Esquema do algoritmo proposto.

Como foi mencionado no Capítulo 4, para gerar a matriz do UniQode é necessário um dicionário. Assim, com a finalidade de usar um dicionário alfanumérico, com 36 (A-Z + 0-9) ou 62 (A-Z + a-z + 0-9) caracteres ou com todos os caracteres imprimíveis pertencentes a uma tabela ASCII (95 caracteres), é possível utilizar os *quanta* 4 ou 5 (Fig. 4.8).

Com isto, a Figura 5.4 mostra uma porção do dicionário usado no algoritmo do UniQode para a mensagem ”Coimbra”.


Figura 5.4: Dicionário usado para a mensagem ”Coimbra”.

Assim, obtém-se os valores referentes a cada pixel da imagem do UniQode, e consoante o valor do pixel for branco ou preto é possível transformá-los em *bits* (0’s ou 1’s) (Fig. 5.5). Deste modo, os bits da imagem do UniQode são agrupados

em conjuntos de 7 *bits* (isto porque com 7 *bits* consegue-se ter todos os caracteres pertencentes à tabela ASCII), e subsequentemente cada conjunto é transformado num carácter ASCII e agrupados numa *string* de caracteres (Fig. 5.6). É importante mencionar que o último carácter da Figura 5.6 é um carácter que não pode ser impresso (não confundir com o carácter do ponto de interrogação). Apesar de não ser possível ver este carácter, os *bits* estão presentes e para o algoritmo de encriptação não é prejudicial.

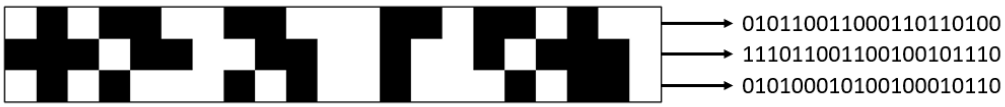



Figura 5.5: Obtenção dos valores dos pixéis e transformação em *bits*.

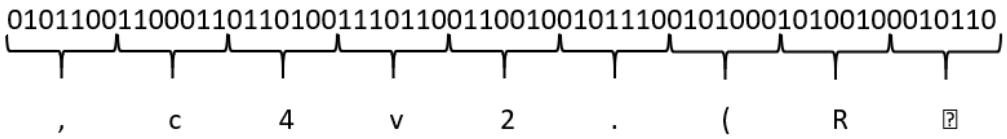



Figura 5.6: Transformação dos *bits* em caracteres ASCII.

Com isto é gerada uma chave de sessão. Uma chave de sessão é uma chave de criptografia simétrica que só vai ser utilizada apenas uma vez para encriptar e desencriptar. Processos de criptografia futuras vão utilizar outra chave de sessão diferente. Outra das características, é o facto de esta chave poder ser de 128, 192 ou de 256 *bits*. Com o fundamento de aumentar a complexidade do sistema, a chave escolhida será a de maior tamanho. Deste modo, o algoritmo de RSA juntamente com a chave pública encriptará a chave de sessão que será usada para o processo de desencriptação.

A chave de sessão gerada vai ser utilizada juntamente com o algoritmo de AES para encriptar a *string* obtida anteriormente, com a finalidade de convertê-la outra vez numa *bitstream* e assim organizar tudo numa matriz binária, que por sua vez vai ser guardada como uma imagem.

O algoritmo seguinte apresenta o pseudo-código das etapas anteriormente descritas.

**Algoritmo 1:** Criação e manipulação dos pixéis do UniQode e respetiva encriptação.

```
1  Função Manipulação_de_pixeis(UniQode):
2      bin_uniqode = ' ';
3      for x = 1,2,...,UniQode.largura do
4          for y = 1,2,...,UniQode.comprimento do
5              pixel = UniQode(x, y);
6              if pixel == 255 then
                   /* Acrescenta um '0' ao final da string          */
7                  bin_uniqode.pushback('0');
8              else
                   /* Acrescenta um '1' ao final da string          */
9                  bin_uniqode.pushback('1');
10     string_uniqode = ConverteToAscii(bin_uniqode);
11     return string_uniqode;
12 Função Encriptação(string, keysize):
13     chave_privada, chave_publica = RSA.generate(keysize);
       /* Chave de sessão de 128/256 bits ou 16/32 bytes           */
14     chave_sessao = random_bytes(32);
       /* Padding à chave pública                                  */
15     cipher_rsa = OAEP.new(chave_publica);
       /* Encriptação da chave de sessão                           */
16     enc_chave_sessao = cipher_rsa.encrypt(chave_sessao);
17     cipher_aes = AES.new(chave_sessao);
       /* Encriptação da mensagem                                  */
18     mensagem_codificada = cipher_aes.encrypt(string);
19     return mensagem_codificada;
20 Função Main:
21     UniQode = UniQode_generation(mensagem1, dicionário);
22     string_uniqode = Manipulação_de_pixeis(UniQode);
23     keysize = 2048;
24     mensagem_codificada = Encriptação(string_uniqode, keysize);
25     informacao_bin = ConvertToBin(mensagem_codificada);
```

Deste modo, utilizando o pseudo-código anterior, é possível encriptar a *string* de caracteres da Figura 5.6 numa *string* encriptada. Como se pode observar pela Figura 5.7 a *string* encriptada tem um maior número de caracteres comparada à original em consequência do processo de *padding* discutido anteriormente. Isto também introduz uma componente aleatória ao método, ou seja, cada vez que se pretender encriptar a *string* de caracteres, o resultado nunca vai ser igual ao anterior.

A Figura 5.8 mostra o resultado da transformação da *string* encriptada em *bits*.

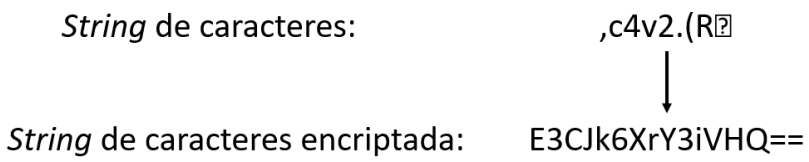



Figura 5.7: *String* de caracteres encriptada.

Estes *bits* são depois organizados numa matriz (Fig. 5.9) e colocados num QR-code com capacidade de conter esta matriz. Os pixéis restantes podem ser utilizados para codificar o tamanho da *bitstream*.

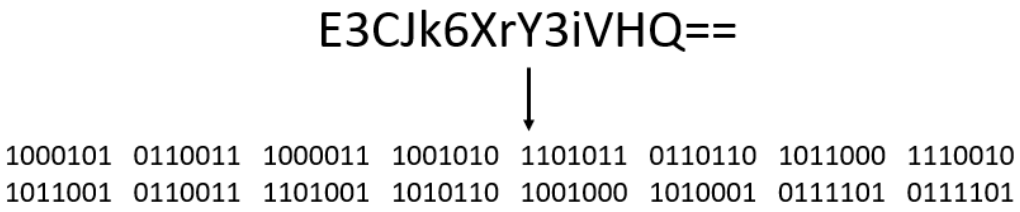



Figura 5.8: Conversão da *string* encriptada para binário.

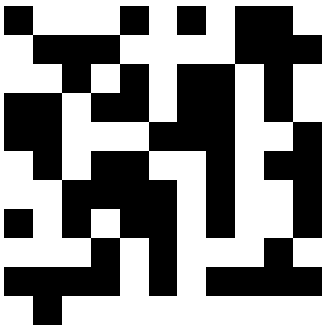


Figura 5.9: *Bits* organizados numa matriz.

A Tabela 5.1 apresenta uma comparação da quantidade de caracteres presentes em mensagens com tamanhos diferentes, da *string* resultante da conversão dos *bits* do UniQode em caracteres ASCII e da *string* encriptada, assim como a percentagem de aumento da *string* encriptada em relação à mensagem original. É possível observar que para uma mensagem com apenas um carácter, a *string* encriptada tem 4 caracteres e para uma mensagem com 402 caracteres, a *string* encriptada tem 692 caracteres, ou seja, 4844 *bits*, o que iria resultar numa matriz de 70x70 pixéis. Isto é benéfico, pois não apresenta um aumento significativo no tamanho depois do processo de encriptação, podendo assim ser encriptada uma mensagem consideravelmente grande.

| Número de caracteres | | | Aumento do número de caracteres da mensagem original para a encriptada (em percentagem) |
|---|---|---|---|
| **Original** | ***String* resultante da conversão dos *bits* do UniQode** | ***String* encriptada** | |
| 1 | 2 | 4 | 300 |
| 7 | 10 | 16 | 129 |
| 23 | 30 | 40 | 74 |
| 26 | 34 | 48 | 85 |
| 50 | 65 | 88 | 76 |
| 77 | 100 | 136 | 77 |
| 131 | 169 | 228 | 74 |
| 185 | 238 | 320 | 73 |
| 402 | 517 | 692 | 72 |

Tabela 5.1: Comparação entre as várias *string* ao longo do processo.

## 5.3 Zonas passíveis a alterações num código QR-code

Após a aplicação da criptografia híbrida ao UniQode (fases A e B), estamos agora em condições de integrar este num código QR-code, no entanto é necessário saber as zonas passíveis a alterações. No Capítulo 4 foi referido que existem zonas pertencentes ao QR-code que são importantes ao bom funcionamento e à descodificação. As zonas que permitem ao *software* de descodificação localizar o QR-code e as zonas pertencentes à indicação da versão e ao tipo de nível de correção de erros não devem ser alteradas. Na Figura 5.10 é exposto o método utilizado para perceber quais as áreas do código sujeitas a modificações.

A Figura 5.10 serve como base para todas as 40 versões do QR-code, ou seja, à entrada teremos um dado QR-code não modificado e à saída temos o tamanho da zona que será modificada sem corromper o bom funcionamento do QR-code original.

Para isso foram divididas em duas fases distintas - a Fase C que é a geração de um determinado QR-code e a Fase D que é o processo de determinação da área passível a alterações.

A Fase C é realizada a partir de uma biblioteca de forma a gerar um QR-code válido e que seja possível descodificar pelos *softwares* existentes nos *smartphones* e computadores.

A Fase D, é composta primeiramente pela exclusão do pixel central, isto é, colocar o respetivo pixel a branco, com o objetivo de tentar encontrar a área interior máxima passível a modificações. Com isto o algoritmo vai aumentar a área remo-

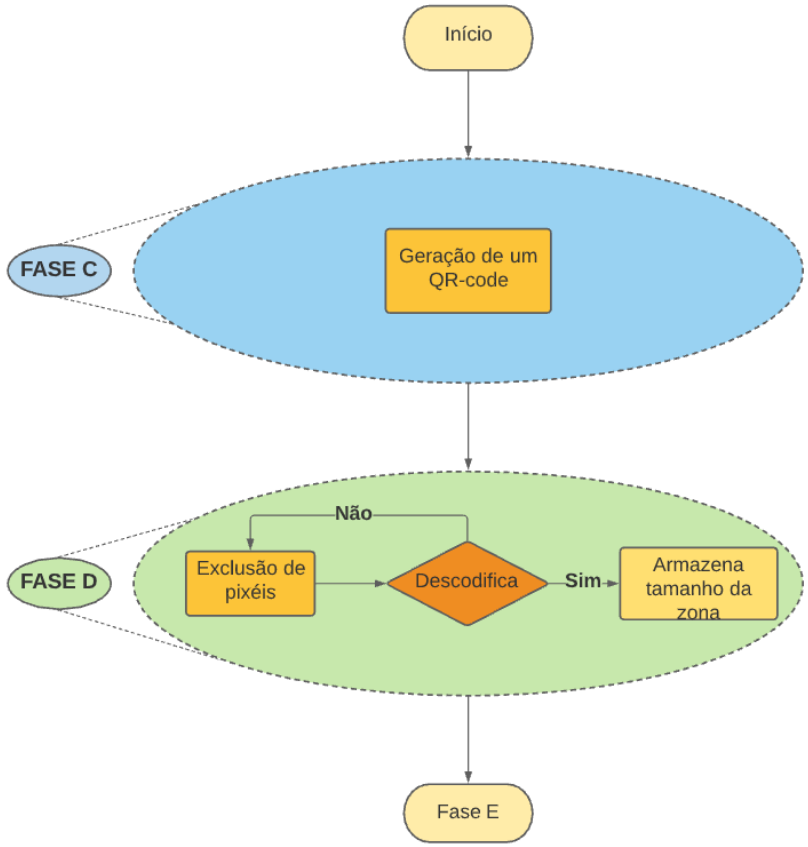



Figura 5.10: Esquema do algoritmo proposto.

vendo os pixéis à volta do pixel removido anteriormente. Se consegue descodificar está encontrado o tamanho máximo da zona sujeita a modificações, se não consegue, aumenta a matriz de pixéis interiores numa casa em todas as direções e tenta assim outra vez a descodificação do QR-code.

Os tamanhos das 40 diferentes zonas são guardados para serem utilizados por fim na fase E onde se integra o UniQode num QR-code, detalhado na subseção seguinte.

O algoritmo 2 apresenta o pseudo-código das etapas anteriormente descritas. É criado um ficheiro que contém o número da versão e a respetiva área que deve ser utilizado na aplicação, com o objetivo de posteriormente conhecer a posição do UniQode.

## 5.4 Integração de um UniQode num QR-code

Uma vez que é conhecido o tamanho das zonas passíveis a alterações dos QR-code, e já se obteve a imagem criptografada através de um método de criptografia híbrida em conjunto com o UniQode, resta assim a integração de ambos os

**Algoritmo 2:** Criação e obtenção das zonas passíveis a alterações do QR-code.

```
1  Função Zonas passíveis a alterações(mensagem2):
2      for versao = 1,2,...,40 do
3          QR-code = QR-code.generation(mensagem2, versao);
4          descodifica = QR-code.descodificacao();
5          i = 0;
6          while descodifica != false do
7              Remove_pixeis();
8              descodifica = QR-code.descodificacao();
9              i = i + 1;
10         Save(i, versao);
```

códigos. O processo está detalhado na Figura 5.11.

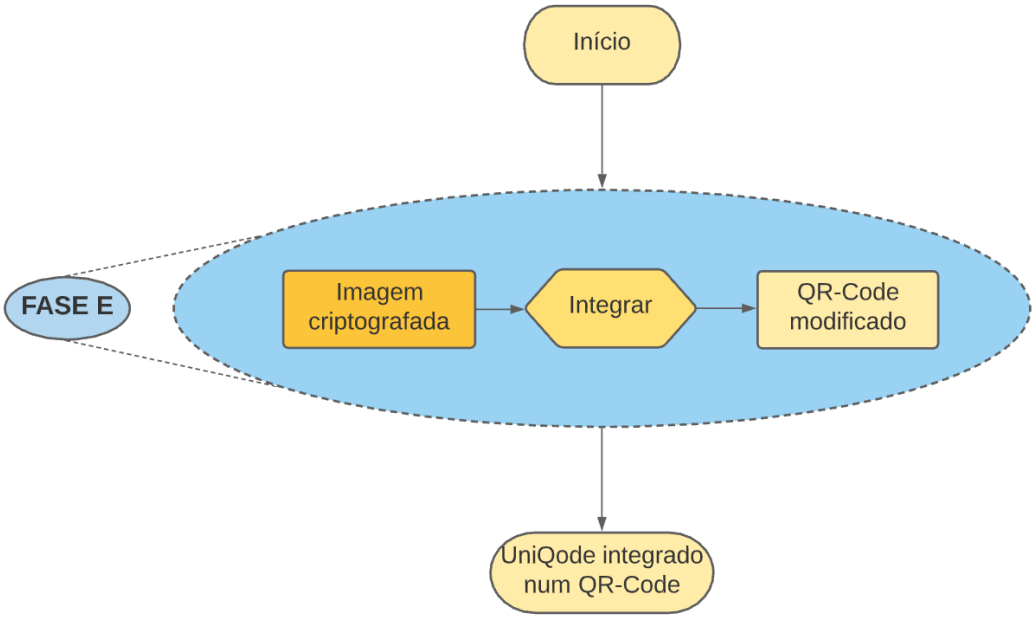



Figura 5.11: Esquema do algoritmo proposto.

A Figura 5.11 refere-se mais uma vez ao processo generalizado, uma vez que, dependendo do tamanho da imagem criptografada, é escolhida uma versão do QR-code que tenha a capacidade de embutir esta imagem na área passiva a modificações, que fora obtida anteriormente. Por fim, é gerada um novo QR-code que contém UniQode cifrado por um modelo de criptografia híbrida.

O modo de funcionamento desta fase consiste no posicionamento dos pixéis referentes ao UniQode codificado na área do QR-code obtida anteriormente. Assim sendo, vai-se obter uma versão do QR-code que tenha uma área suficientemente

grande, de forma a colocar os pixéis do UniQode. Se por acaso for gerada um QR-code grande por causa da dimensão da mensagem em que a zona passível a modificações é superior à dimensão da matriz do UniQode codificado, o restante espaço pode ser preenchido com os pixéis do QR-code que anteriormente lá se encontravam ou por outra informação adicional.

A Figura 5.12 mostra o UniQode encriptado da Figura 5.9 embutido num QR-code. Como se pode observar, este QR-code não apresenta diferenças visuais para um QR-code normal.

Figura 5.12: UniQode encriptado integrado num QR-code.

De seguida será exposta a validação deste novo método, que permite a implementação de um modelo de criptografia em códigos gráficos bidimensionais.

## 5.5 Validação do Método Proposto

Para a validação deste método de leitura ótica, foi desenvolvida uma aplicação em *iOS*, garantido assim a facilidade de leitura através de um *smartphone*.

Este software será capaz de ler os dois códigos gráficos e através de uma *API* externa desencriptar a imagem. Na Figura 5.13 pode-se ver um exemplo ilustrativo da leitura destes dois códigos usando um dispositivo *iOS*.

O conteúdo destas imagens só é devidamente disponibilizado se o proprietário possuir as devidas chaves de desencriptação. Caso um outro utilizador queira ter acesso à informação através da aplicação este só consegue ter acesso ao conteúdo do QR-code, uma vez que não possui a chave de desencriptação da informação referente ao UniQode.

Figura 5.13: Aplicação de leitura do novo código ótico no sistema operativo iOS.

Este método torna-se assim vantajoso para a autenticidade do produto, pois é preciso sempre ter acesso às duas chaves para que a verificação e validação do mesmo seja bem realizada.

A aplicação é constituída por uma biblioteca pertencente ao *OpenCV*, que deteta o QR-code e retorna a matriz rectificada. Com esta matriz, vai ser possível obter os pixéis relativos à área do UniQode e assim encaminhar para a *API*. A *API* na sua vez vai desencriptar a mensagem e enviar para a aplicação de modo a ser observada pelo utilizador (Fig. 5.14).

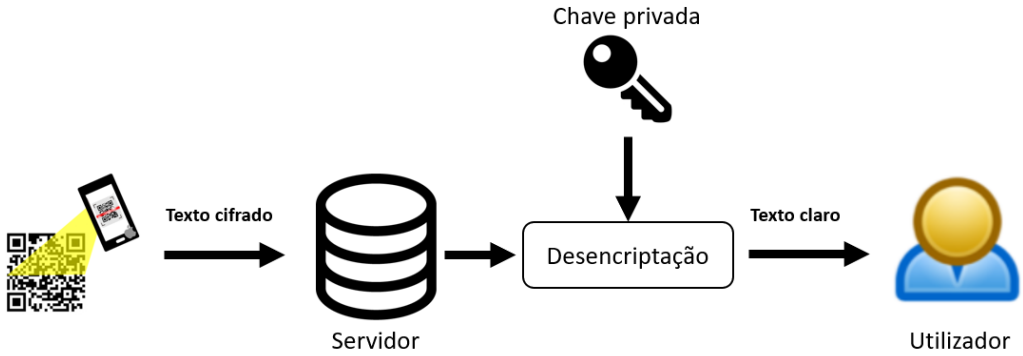



Figura 5.14: Exemplo do processo de desencriptação.

Tendo em conta o algoritmo 3 descrito posteriormente, com a ajuda de uma câmara de um *smartphone* vai ser possível à aplicação detetar um QR-code. O algoritmo do *OpenCV* vai retornar a matriz do QR-code, e com esta matriz e as zonas anteriormente obtidas na subsecção 5.3 vai ser possível obter a sub-matriz correspondente ao UniQode encriptado. Esta matriz é transformada em *bits* e é

**Algoritmo 3:** Descodificação

```
1  Função Desencriptação(string_binaria, chave_privada):
       /* Converte a string binária numa string de caracteres.        */
2      string = converte_string(string_binaria);
       /* Processo de desencriptação.                                 */
3      cipher_rsa = OAEP.new(chave_privada);
4      chave_sessao = cipher_rsa.decrypt(enc_chave_sessao);
5      cipher_aes = AES.new(chave_sessao);
6      string_descodificada = cipher_aes.decrypt(string);
       /* Converte a string_descodificada novamente numa string binária.
          */
7      string_binaria = converte_binary(string_descodificada);
8      return string_binaria;
9
10 Função Descodifica_UniQode(string_binaria, dicionário):
       /* Vai organizar a string_binaria numa matriz de forma a exibir a
          mensagem original.                                          */
11     matriz = ordenacao_pixeis(string_binaria);
12     mensagem = descodifica(matriz, dicionario);
13     return mensagem;
14
15 Função Main:
       /* A função do OpenCV deteta um QR-code e retorna a matriz
          rectificada.                                                */
16     QrCode = OpenCV_QRDetector();
17     UniQode_encriptado = recolhepixeis(QrCode, posição);
       /* Converte os pixéis em binário.                              */
18     bin_UniQode = converte_binary(UniQode_encriptado);
       /* Envia os bits para o servidor.                              */
19     string_binaria = Desencriptação(bin_uniqode, chave_privada);
20     mensagem = Descodifica_UniQode(string_caracteres, dicionário);
```

enviada para o servidor juntamente com a chave privada do utilizador, onde vai ser convertida em caracteres ASCII, ou seja, num texto cifrado e por sua vez vai ser desencriptada. O texto claro é novamente convertido em *bits* e remetido para o *smartphone* do utilizador onde vai ser organizada numa matriz de forma a que, juntamente com o dicionário do UniQode, seja possível obter a mensagem original.

# 6

# Resultados e discussão

Este capítulo apresenta os resultados obtidos ao longo desta dissertação a fim de comprovar o método realizado.

A avaliação foi realizada por meio de uma aplicação para *iOS* de forma a demonstrar o desempenho da leitura e descodificação dos códigos de leitura ótica. Esta aplicação deve manter a rapidez de deteção e descodificação que se vê nas tradicionais aplicações para códigos de leitura ótica.

Portanto, o método que foi criado deve manter as mesmas vantagens que um QR-code normal, nomeadamente a rapidez de leitura e de criação, mantendo também os mesmos atributos visuais e envolvendo uma identificação rápida usando um *smartphone*.

Como foi mencionado no Capítulo 4, existem vários níveis para o algoritmo de correção de erros do QR-code. Este algoritmo faz com que seja possível recuperar informação, caso o QR-code esteja sujo ou danificado. Para entender qual o melhor nível a utilizar nesta solução foi utilizada uma biblioteca criada por Nayuki[1] de forma a produzir QR-code válidos. Esta biblioteca foi escolhida porque apresenta um código simples de compreender e de uma forma fácil é possível alterar configurações de um modo intuitivo.

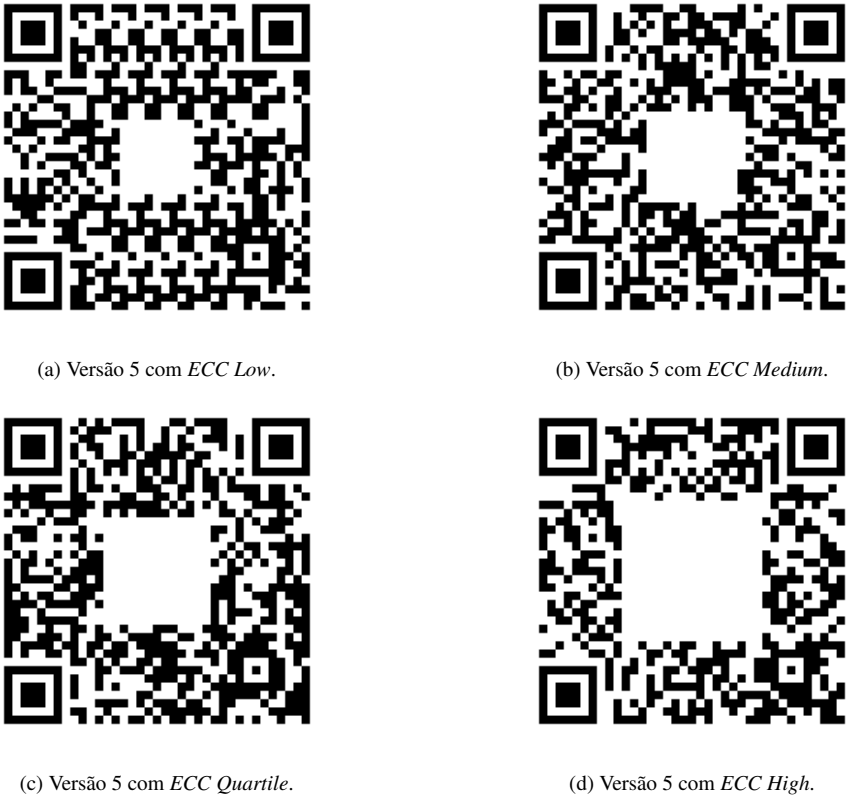
Com isto, foi criado um código adaptado a partir desta biblioteca de forma a obter o nível de correção de erros (L, M, Q, H) que seja o mais adequado para a integração do UniQode. Para isso começou-se por retirar fragmentos ao QR-code para cada nível até que já não fosse possível descodificar a informação. Como se pode observar pela Figura 6.1, existem diferentes tamanhos para os quatro níveis de correção de erros existentes para a mesma versão. Isto era de esperar pois cada nível consegue reparar uma percentagem diferente de dados, como referido anteriormente.

Como o objetivo é ter a maior capacidade possível de forma a embutir o UniQode codificado, foi escolhido o nível de correção *High*, pois é o que apresenta maior correção de erros. Apesar de poder não ser necessário utilizar todos os pixéis para o UniQode, este é o mais vantajoso pois tem maior capacidade para embutir um superior número de pixéis.

Assim, com o nível de correção de erros a *High*, foi-se obter os limites máximos de pixéis para as 40 versões do QR-code, obtendo assim a Tabela 6.1.

Para esta tabela, os limites foram obtidos a partir da descodificação correta do QR-code utilizando uma aplicação nativa de um *smartphone*. Os valores só foram considerados se a aplicação reconhecesse e descodificasse corretamente o QR-code

[1] `https://github.com/nayuki/QR-Code-generator`



(a) Versão 5 com *ECC Low*.

(b) Versão 5 com *ECC Medium*.

(c) Versão 5 com *ECC Quartile*.

(d) Versão 5 com *ECC High*.

Figura 6.1: Diferentes zonas para a versão 5 com os diferentes níveis de correção de erros.

utilizando o método aplicado em 5.3.

Como o UniQode é constituído por pixéis pretos e brancos, foi embutido uma matriz aleatória nas respetivas zonas do QR-code com o propósito de verificar se era possível obter o conteúdo da mensagem. Assim, a Figura 6.2 mostra uma matriz aleatória utilizando os valores da Tabela 6.1. Isto mostra que é possível descodificar o QR-code mesmo que os seus módulos internos sejam alterados.

De referir que as versões que tenham bastantes módulos (versão 25, 35 e 40 da Figura 6.2) devem ser impressas ou criadas com um maior número de pixéis por módulo, de forma a ser possível ser descodificado e identificado como um QR-code, o que apresenta uma desvantagem quando se pretende aplicar num local que não dispõe de muito espaço. Isto acontece porque o *hardware* de descodificação pode não ter resolução suficiente, de forma a identificar todos os módulos do QR-code.

Com isto foi gerada a matriz relativa ao UniQode. Esta matriz pode ter diversos formatos, como mostra a Figura 6.3. Como referido no Capítulo 4, na matriz quadrada nem todas as células pertencem ao dicionário, pois a mensagem é de menor dimensão que a matriz. Estas células não vão ser consideradas no processo de

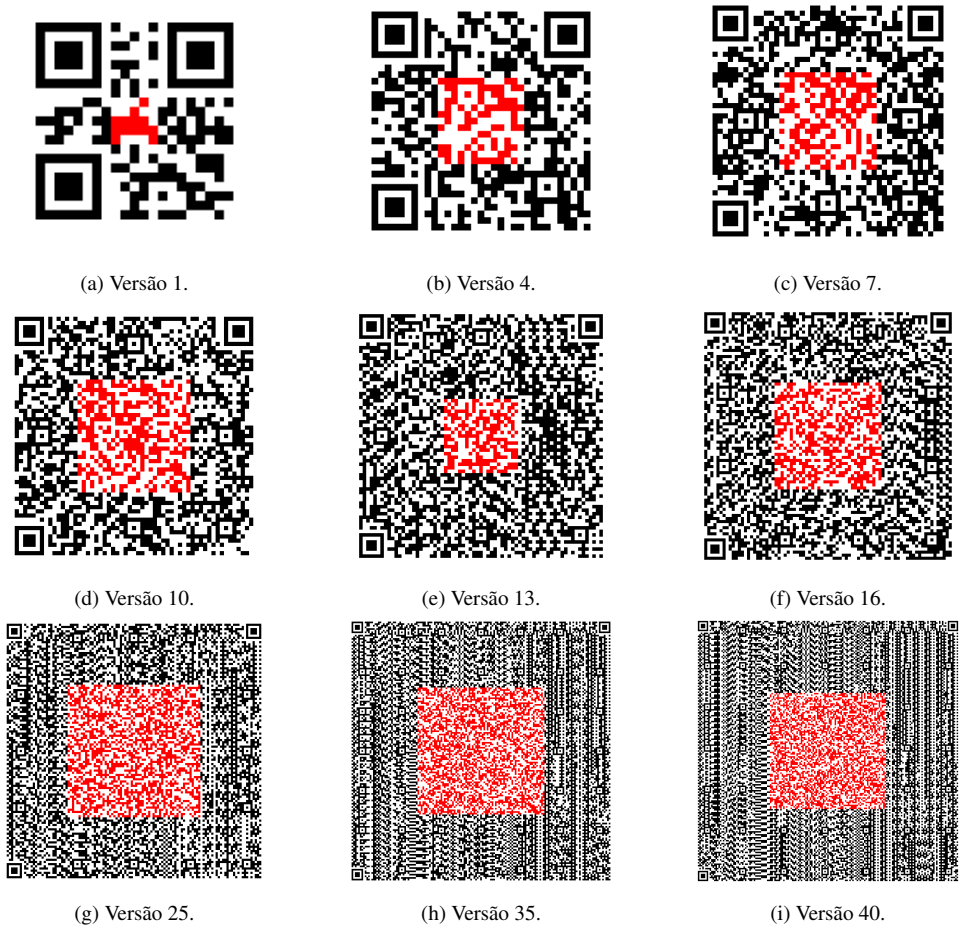
| Versão | Dimensão do QR-code | Limite de pixéis para o UniQode | Versão | Dimensão do QR-code | Limite de pixéis para o UniQode |
|---|---|---|---|---|---|
| *1* | 21x21 | 5x5 | *21* | 101x101 | 39x39 |
| *2* | 25x25 | 9x9 | *22* | 105x105 | 49x49 |
| *3* | 29x29 | 11x11 | *23* | 109x109 | 55x55 |
| *4* | 33x33 | 13x13 | *24* | 113x113 | 57x57 |
| *5* | 37x37 | 17x17 | *25* | 117x117 | 61x61 |
| *6* | 41x41 | 17x17 | *26* | 121x121 | 61x61 |
| *7* | 45x45 | 19x19 | *27* | 125x125 | 65x65 |
| *8* | 49x49 | 21x21 | *28* | 129x129 | 61x61 |
| *9* | 53x53 | 23x23 | *29* | 133x133 | 69x69 |
| *10* | 57x57 | 27x27 | *30* | 137x137 | 71x71 |
| *11* | 61x61 | 29x29 | *31* | 141x141 | 71x71 |
| *12* | 65x65 | 31x31 | *32* | 145x145 | 69x69 |
| *13* | 69x69 | 21x21 | *33* | 149x149 | 77x77 |
| *14* | 73x73 | 33x33 | *34* | 153x153 | 77x77 |
| *15* | 77x77 | 29x29 | *35* | 157x157 | 77x77 |
| *16* | 81x81 | 35x35 | *36* | 161x161 | 77x77 |
| *17* | 85x85 | 39x39 | *37* | 165x165 | 85x85 |
| *18* | 89x89 | 33x33 | *38* | 169x169 | 81x81 |
| *19* | 93x93 | 41x41 | *39* | 173x173 | 75x75 |
| *20* | 97x97 | 39x39 | *40* | 177x177 | 79x79 |

Tabela 6.1: Tabela com as dimensões do QR-code e as respetivas zonas para as 40 versões.

descodificação do UniQode, mas sim na encriptação.

A Figura 6.4 mostra o resultado final depois da encriptação do UniQode da Figura 6.3. Apesar das matrizes iniciais (Fig. 6.3) serem diferentes em tamanho e o número de pixéis ser consideravelmente diferente, 207 comparado com 625 pixéis, as matrizes finais não são muito diferentes, 289 pixéis (Fig. 6.4a) comparado a 324 (Fig. 6.4b).

Para a encriptação utilizou-se a criptografia híbrida, o algoritmo de AES juntamente com o de RSA, porque ao utilizar só o método de RSA juntamente com o *padding*, a mensagem cifrada iria ser igual ao tamanho da chave utilizada, que no nosso caso foi de 2048 *bits*. Ou seja, para uma chave de 2048 *bits*, iríamos ter um texto cifrado de 2048 *bits*. Isto iria ser desvantajoso pois o limite de pixéis a utilizar iria ser de $\sqrt{2048} \approx 46$, isto significa que iria ser preciso uma área com 46x46 pixéis, que só a versão 22 e superiores tinham esta capacidade. Desta forma o QR-code a utilizar iria ser muito grande, o que não se pretende na maioria dos



(a) Versão 1. (b) Versão 4. (c) Versão 7.

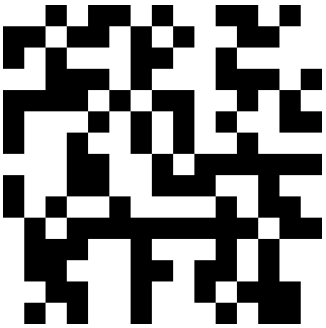
(d) Versão 10. (e) Versão 13. (f) Versão 16.

(g) Versão 25. (h) Versão 35. (i) Versão 40.

Figura 6.2: Algumas versões do *QR-code* com pixéis aleatórios a vermelho.

casos.

(a) Matriz de 1x23 células.

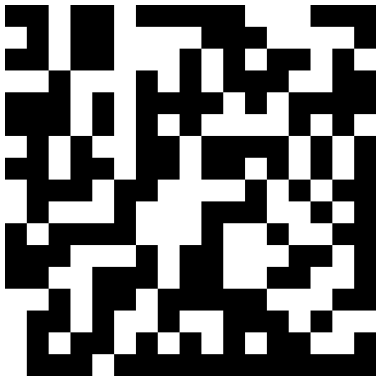


(b) Matriz de 5x5 células.

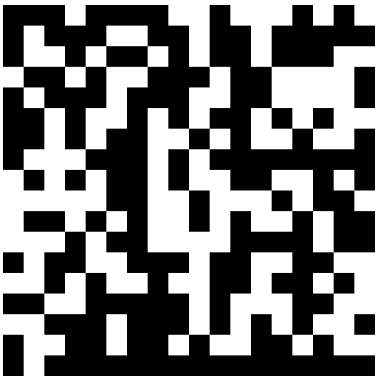
Figura 6.3: UniQode produzido com a mensagem ”Universidade de Coimbra”.

Assim, para a mensagem utilizada na criação dos UniQode da Figura 6.3, que tem um tamanho de 23 caracteres, o QR-code ideal iria ser a versão 5 para a primeira

configuração e a versão 7 para a segunda. Usando só um carácter na mensagem inicial do UniQode, o resultado final do UniQode encriptado vai ser uma matriz 6x6 (Fig. 6.5), que pela observação da Tabela 6.1 só a versão 2 permitia a introdução desta matriz.



(a) UniQode encriptado a partir da Fig.6.3a.



(b) UniQode encriptado a partir da Fig.6.3b.

Figura 6.4: Resultado do UniQode encriptado com a criptografia híbrida.

Figura 6.5: Exemplo de um UniQode encriptado com um único caracter.

Para a descodificação do QR-code com o UniQode, foi desenvolvida uma aplicação usando uma função do *OpenCV*, denominada de QRCodeDetector. Esta função tem uma característica importante, pois devolve a matriz de pixéis, assim como a mensagem pertencente ao QR-code. A matriz de pixéis é importante, pois com ela conseguimos obter os pixéis referentes ao UniQode. Porém, esta biblioteca não consegue reconhecer versões do QR-code superiores à versão 5, constituindo assim uma desvantagem, pois ficamos limitados ao conjunto das versões 2 a 5. Isto, deve-se provavelmente a um problema na gestão de memória que persistiu até à última versão disponível no momento em que esta dissertação foi escrita. Assim sendo, ficámos limitados até à versão 6 do QR-code.

Para conhecer a localização do UniQode, fomos à matriz que a função do *OpenCV* retornou e com a Tabela 6.1 obteve-se a matriz correspondente ao

UniQode. Como mencionado anteriormente, os pixéis foram convertidos em *bits*, e por sua vez, num texto ASCII. Este texto, através da API, foi desencriptado utilizando a chave privada fornecida pelo utilizador. Como se pode constatar na Figura 6.6 ou na ligação[2] em rodapé é possível obter os dois códigos bidimensionais, assim como as duas informações, constatando assim, que este método funciona como pretendido.

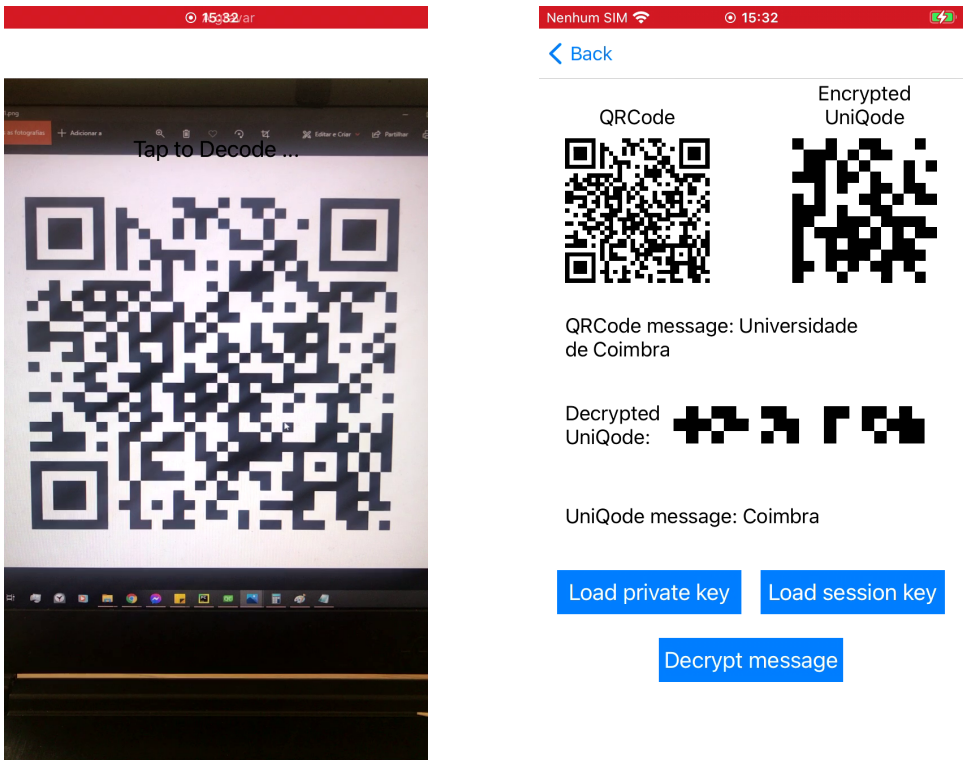



Figura 6.6: Exemplo da aplicação.

[2] https://vimeo.com/531215767

# 7

# Conclusão

## Conteúdo

A utilização exponencial de códigos gráficos bidimensionais por parte das empresas tem sido uma ferramenta de marketing popular, onde toda a informação é acedida facilmente e de uma forma eficaz por parte de qualquer consumidor através de um dispositivo móvel. Para além de assegurar a disponibilidade da informação, têm que ser garantido a segurança e a autenticidade da mesma, impedindo que um dado produto seja contrafeito. Todas estas características são encontradas no UniQode que através de um dicionário próprio e utilizando criptografia simétrica consegue a autenticidade de um determinado produto.

Com o objetivo de aliar esta segurança dada pelo UniQode à grande capacidade de armazenamento de informação do QR-code, foi definida uma metodologia para que o UniQode fosse integrado no QR-code. Neste trabalho, primeiramente foram estudadas e determinadas zonas passiveis a alterações no QR-code, para que o UniQode conseguisse ser embutido nesta nova zona sem corromper os dados no resto do QR-code.

Nesta dissertação, após a validação dessas zonas, foi então definida uma metodologia para que se pudesse aliar o melhor dos dois códigos de leitura ótica, em que através de criptografia híbrida à imagem gerada pelo UniQode ela pudesse ser embutida dentro das zonas passiveis de alteração no QR-code.

Com o intuito de validar esta metodologia foi adaptada uma aplicação em iOS, onde foi possível aceder facilmente e de uma forma eficaz à informação do produto. Nesta aplicação através do uso de chaves privadas é possível aceder à informação contida no UniQode, o que se torna vantajoso para que um utilizador possa autenticar e validar um dado produto.

## 7.1 Trabalhos Futuros

A metodologia definida nesta dissertação em conjunto com a aplicação adaptada para iOS serão a base para projetos futuros.

Uma vez que o número de utilizadores do sistema operativo Android é significativo pode também ser realizada uma adaptação da aplicação para este sistema, em que o alcance desta metodologia terá uma maior visibilidade.

Um trabalho futuro também poderá ser o desenvolvimento de uma aplicação de raiz em ambos os sistemas operativos, para que esta possa suportar todas as versões do QR-code, uma vez que a aplicação atual só suporta até à versão 5.

# Bibliografia


[1] D. Waves, "Qr code essentials." [Online]. Available: https://delivr.com/resources/files/1058/DENSO_ADC_QR_Code_White_Paper.pdf

[2] Y.-W. Chow, W. Susilo, G. Yang, J. G. Phillips, I. Pranata, and A. M. Barmawi, "Exploiting the error correction mechanism in qr codes for secret sharing," in *Information Security and Privacy*, J. K. Liu and R. Steinfeld, Eds. Springer International Publishing, 2016.

[3] L. Cruz, B. Patrão, and N. Gonçalves, "Graphic code: A new machine readable approach," in *2018 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*. IEEE, 2018, pp. 169–172.

[4] J. Cucurull and S. Guasch and A. Escala and G. Navarro-Arribas and V. Acín, "Qr steganography: A threat to new generation electronic voting systems," in *2014 11th International Conference on Security and Cryptography (SECRYPT)*, 2014, pp. 1–8.

[5] M. Naor and A. Shamir, "Visual cryptography," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1994.

[6] S. Jiao, J. Feng, Y. Gao, T. Lei, and X. Yuan, "Visual cryptography in single-pixel imaging," *Optics express*, vol. 28, no. 5, 2020.

[7] S. Kamal and B. Ameen, "A new method for ciphering a message using qr code," *Computer Systems Science and Engineering*, vol. 6, 06 2016.

[8] L. Cruz, B. Patrão, and N. Gonçalves, "Graphic code - a new reliable machine system for coding. unpublished technical report. university of coimbra."

[9] H.-L. Lee and L.-H. Chen, "A novel printable watermarking method in dithering halftone images," *Advances in Multimedia*, vol. 2016, 2016.

[10] Y. Guo, O. C. Au, R. Wang, L. Fang, and X. Cao, "Halftone image watermarking by content aware double-sided embedding error diffusion," *IEEE Transactions on Image Processing*, vol. 27, 2018.

[11] K. Goyal and S. Kinger, "Modified caesar cipher for better security enhancement," *International Journal of Computer Applications*, vol. 73, no. 3, pp. 0975–8887, 2013.

[12] M. Rivain, "On the physical security of cryptographic implementations," Ph.D. dissertation, University of Luxembourg, Luxembourg, Luxembourg, 2009.

[13] F. A. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Attacks on copyright marking systems," in *International workshop on information hiding*. Springer, 1998.

[14] D. F. Pigatto, "Segurança em sistemas embarcados críticos-utilização de criptografia para comunicação segura," Ph.D. dissertation, Universidade de São Paulo, 2012.

[15] FIPS-197: Advanced Encryption Standard (AES), Federal Information Processing Standards Publication, National Institute of Standards and Technology (NIST), November 2001. [Online]. Available: https://doi.org/10.6028/NIST.FIPS.197

[16] J. Daemen and V. Rijmen, "The rijndael block cipher: Aes proposal," in *First candidate conference (AeS1)*, 1999.

[17] Cristina Caldeira & Pedro Quaresma, "Códigos e criptografia," notas de leitura da disciplina de Códigos e Criptografia do Departamento de Matemática da Universidade de Coimbra.

[18] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, 1978.

[19] L. Zou, M. Ni, Y. Huang, W. Shi, and X. Li, "Hybrid encryption algorithm based on aes and rsa in file encryption," in *Frontier Computing*, J. C. Hung, N. Y. Yen, and J.-W. Chang, Eds. Springer Singapore, 2020.

[20] "Gs1, página institucional." [Online]. Available: https://www.gs1pt.org/

[21] S. El Assad, M. Farajallah, and C. Vladeanu, "Chaos-based block ciphers: An overview," in *2014 10th International Conference on Communications (COMM)*, 2014.

[22] S. Chirakkarottu and S. Mathew, "A novel encryption method for medical images using 2d zaslavski map and dna cryptography," *SN Applied Sciences*, 2020.

[23] D. Boneh, A. Joux, and P. Q. Nguyen, "Why textbook elgamal and rsa encryption are insecure." Berlin, Heidelberg: Springer-Verlag, 2000.

[24] J. Katz and Y. Lindell, *Introduction to modern cryptography*. CRC press, 2020.

[25] V. Shoup, "Oaep reconsidered," *Journal of Cryptology*, vol. 15, no. 4, pp. 223–249, 2002.

[26] D. Mudzingwa and R. Agrawal, "A study of methodologies used in intrusion detection and prevention systems (idps)," in *2012 Proceedings of IEEE Southeastcon*. IEEE, 2012.

[27] J. Knockel, T. Ristenpart, and J. Crandall, "When textbook rsa is used to protect the privacy of hundreds of millions of users," *arXiv preprint arXiv:1802.03367*, 2018.

[28] H.-L. Lee and L.-H. Chen, "A novel printable watermarking method in dithering halftone images," *Advances in Multimedia*, vol. 2016, 2016.