Universidade da Beira Interior
Faculdade de Engenharia
Departamento de Informática

Propostas para Trabalhos de Grupo
Team Work Proposals

Segurança de Sistemas Informáticos
Computer Systems Security

Departamento de Informática
Department of Computer Science
Universidade da Beira Interior
University of Beira Interior

Pedro R. M. Inácio
inacio@di.ubi.pt
2024/25

Conteúdo

1 Introdução

Introduction

As seguintes propostas de trabalho foram elaboradas no contexto da unidade curricular de Segurança de Sistemas Informáticos do curso de mestrado em Engenharia Informática da Universidade da Beira Interior, e servem primariamente o propósito de melhorar a destreza dos seus executantes no que diz respeito ao desenho e desenvolvimento de sistemas de software em geral, à inclusão de aspetos de segurança durante a fase de projeto dessas aplicações, ao manuseamento e correta utilização de mecanismos da criptografia, e ao teste do sistema final em particular. Como tal, estas propostas não replicam necessariamente, ou em todo o detalhe, aplicações das tecnologias actualmente em uso.

The following proposals were elaborated within the context of the subject of Computer Systems Security of the masters degree on Computer Science and Engeneering of the University of Beira Interior, and their primary purpose it to improve the dexterity of the students concerning the design and development of software systems in general, the inclusion of security aspects during the engineering phase of those applications and the handling and correct utilization of cryptographic mechanisms, and to the testing of the resulting system in particular. As such, they may not all replicate real life applications of nowadays technologies.

Note também que os estudantes são livres de submeter uma nova proposta, desde que esteja em concordância com os objetivos da unidade curricular. Adicionalmente, a proposta deverá ser discutida com o regente antes de ser aceite como um tópico de trabalho de equipa válido.

Notice also that the student is free to submit a new proposal, as long as it is in accordance with the objectives of this subject. Additionally, the proposal has to be discussed with the Professor prior to being accepted as a valid teamwork topic.

Como de resto discutido durante as aulas, as equipas devem ter entre quatro e cinco elementos, e a implementação destas propostas deve ser complementada com artefactos que provem a autoria e originalidade (e.g., mostrando evidências do uso de ferramentas de gestão do projeto de software) e com uma apresentação em que se discutem as dificuldades encontradas ao longo do trabalho e se justificam todas as escolhas tomadas. A apresentação ocorre no final do semestre, recorrendo a um pequeno conjunto de diapositivos.

As discussed in the classes, the teams should be formed by four or five students, and the implementation of these proposals should be complemented with artifacts that prove the authoring and originality (e.g., showing evidences of the usage of software management tools) and with a presentation in which the difficulties faced along the work are discussed and the choices that have been taken are justified. The presentation will take place at the end of the semester, resorting to a small set of slides.

As aplicações podem ser implementadas recorrendo a qualquer estúdio ou ambiente de desenvolvimento integrado, sendo esse pormenor deixado ao critério do estudante. São também livres na escolha da plataforma alvo ou tecnologias *web* envolvidas, se aplicáveis, a não ser que os meios e tecnologias a utilizar estejam diretamente indicados na proposta). Só terão de ser levados em conta os dois apontamentos que se seguem: (i) não presuma nem esteja à espera que o Professor seja capaz de responder a todas as questões específicas às tecnologias que escolheu (por exemplo, não espere que ele saiba como trabalhar simultaneamente com o Netbeans, Eclipse, Android Studio, Xcode, etc.) e, (ii) a aplicação deve estar a funcionar corretamente na data de entrega, independentemente das ferramentas ou tecnologias utilizadas. Alguns dos problemas que a equipa encontrar durante o desenvolvimento destes projetos podem já ter sido tratados ou resolvidos nas aulas, e é livre de reutilizar, se aplicável, qualquer pedaço de código ou recurso desenvolvido ao longo das mesmas.

The applications can be implemented resorting to any development studio or integrated development environment of your choice. The team is also free to decide the programming language or the web technology, if applicable, that is going to be used in the development of the project, unless the means or technologies that should be used are directly specified in the proposal. Nonetheless, the following two remarks should be taken into consideration: (i) do not expect the Professor to be able to answer

to all technology specific questions (e.g. do not expect him to know how to simultaneously work with Netbeans, Eclipse, Visual Studio, Android Studio, Xcode, etc.) and, (ii) the application should be correctly functioning by the time it is delivered. Some of the problems the team may face during the development of these projects may have been already addressed during the practical classes and, as such, the team is free to reuse, if applicable, any code developed along those classes.

É da responsabilidade dos estudantes a pesquisa de detalhes específicos à solução dos problemas propostos, assim como de maneiras de testar a validade dessas soluções. Para além de referirem a solução do problema na apresentação, os estudantes deve também descrever o modo como validaram o que foi feito (se aplicável) e incluir alguma teoria de como foi conseguido.
It is of the responsibility of the students to search for specific details concerning the proposal at hands and for ways to validate the solutions. Besides including the solution to the problem in the presentation, students should also describe the means used to validate the work (if applicable) and include some of the theory behind what was done.

A apresentação deve conter pelo menos um diapositivo (e.g., Engenharia de Software e Segurança) em que são discutidos claramente os pontos relativos à segurança tidos em conta durante a fase de desenho e projeção da aplicação ou sistema. Pode inclusivamente conter uma modelação prévia dos ataques a que a aplicação ou sistema desenvolvido pode estar afeto ou endereça (esta modelação é alvo de estudo numa aula – possivelmente a aula 8 ou 9 – desta unidade curricular), bem como a descrição dos testes feitos para garantir que a aplicação cumpria os objetivos de segurança. É obrigatória a inclusão do modelo do sistema, do modelo de ataque e das propriedades que devem ser garantidas, bem como uma discussão de como é que o sistema desenvolvido as garante.
The presentation shall contain one or more slides (e.g., Security and Software Engineering) for clearly discussing security related aspects that were taken into account during the design and projection phase of the application. It should also include models for the attacks to which the application or system may be exposed to or that the implementation addresses (this modeling is the subject of study in one of the lectures – probably lecture 8 or 9 – of this course unit), as well as a description of the tests that were performed to ensure that the application meets the security objectives. It is mandatory to include the system and threat models, and the security properties that should assured by design, along with a discussion on how does the system assures those properties.

1.1 Entrega do Trabalho
Delivery of the Works

A entrega do trabalho é feita única e exclusivamente via *moodle* até às 23:55 do dia de entrega do trabalho e os **nomes dos ficheiros devem seguir a especificação** incluída na secção dedicada a essa entrega na área da unidade curricular naquela plataforma. Por cada dia de atraso na entrega de qualquer elemento do trabalho (código de implementação, *scripts* de instalação ou outros artefactos), descontam-se 0,5 valores (aos 5).

1.2 Sugestão de Alinhamento para a Apresentação
Suggested Lineup for the Presentation

Como dito anteriormente, a defesa do trabalho deve ser acompanhada por um breve conjunto de diapositivos (nunca mais do que 10). A apresentação deve rondar os 10 minutos. Devem considerar fazer um conjunto de 10 diapositivos para guiar o discurso com o seguinte alinhamento (façam as adaptações que considerem necessárias):

- 1 diapositivo com o titulo do trabalho e elementos do grupo;
- 1 diapositivo com os objetivos do trabalho;
- 1 diapositivo dedicado ao levantamento de requisitos de segurança;
- 1 diapositivo dedicado à modelação do sistema e de ataque;

- 1 diapositivo dedicado(s) à implementação;
- 1 diapositivo dedicado à apresentação do sistema/aplicação desenvolvida (este diapositivo é só para lembrar para fazer o *switch* para um demonstrador real ou para um video da aplicação a correr);
- 1 diapositivo a referir os testes efetuados;
- 1 diapositivo com a análise crítica;
- 1 diapositivo dedicado aos objetivos alcançados / conclusões e trabalho futuro.
- 1 diapositivo a dizer `Bem haja pela atenção. Perguntas?`

A descrição das propostas é bastante breve e por vezes desprovida de detalhes, para que possam procurar alguma informação junto dos docentes da unidade curricular e, simultaneamente, analisar e decidir sobre as soluções alternativas.

2 Moeda Criptográfica Controlada Centralmente (*codenamed*: fr€coin)

2.1 Breve Introdução

O objetivo principal deste projeto é o de criar uma moeda criptográfica. Ao contrário da BitCoin, esta moeda deve ser controlada centralmente, i.e., deve existir um sistema central (Entidade Central) que controla a quantidade de moeda existente e todos os gastos. Quando um novo utilizador se regista, deve ser gerado um par de chaves (e.g., RSA), cuja chave pública identifica as suas transações no sistema. Se um utilizador quiser enviar dinheiro a outro utilizador, deve criar um documento em que coloca a sua chave pública, a quantidade de freecoins que está a enviar e a chave pública do destinatário. O documento é assinado pelo emissor e enviado para a Entidade Central, que verifica se a transação se pode dar (i.e., se o emissor existe e tem dinheiro suficiente e se o destinatário existe). Caso a transação possa ser feita, assina também o documento e envia-o para o destinatário. Esta entidade central vai construindo um caderno de transações com todas as transações criadas até à data. Este caderno só é conhecido por essa entidade. Cada utilizador só deve conhecer (de forma legítima) as transações que efetuou ou as que refletem dinheiro que recebeu.

A criação da moeda deve ser feita como se sugere a seguir. A cada 30 segundos, a entidade central vai atribuir 1 fr€coin a quem lhe fizer lhe resolver primeiro um desafio. O desafio é simples: a entidade central gera um valor aleatório com b bits e envia esse valor para a rede. O primeiro utilizador que lhe enviar um ficheiro cujos primeiros b bits do valor de *hash* do SHA256 são iguais aos do desafio, ganha a moeda.

Pensem numa forma de atacar o sistema (uma falha da sua implementação) e dediquem-lhe um pequeno intervalo de tempo na apresentação.

2.2 Funcionalidades Básicas

- O registo de um novo utilizador é feito de forma segura;
- Os pares de chaves são gerados do lado do utilizador;
- Todas as comunicações entre utilizadores e entidade central são feitas de forma segura (e.g., cifradas e protegidas por mecanismos de integridade – usar *Advanced Encryption Standard* (AES) e *Hash-based Message Authentication Codes* (HMACs));

- O sistema permite as transações conforme descritas na breve descrição;
- O sistema implementa a funcionalidade de geração de novas moedas.

2.3 Funcionalidades Avançadas

- O sistema suporta transações completamente anónimas entre utilizadores (i.e., geração de um par de chaves por cada transação);
- O sistema suporta autenticação forte mútua (cliente **e** servidor);
- A entidade central controla a emissão da moeda, ajustando o grau de dificuldade do problema à velocidade com a rede o resolve;
- A entidade central é também uma autoridade certificadora, e o sistema passa a estar suportado por uma infrastrutura de chave pública;
- O sistema é implementado com criptografia sobre curvas elípticas;
- Outras funcionalidades relevantes no contexto da segurança do sistema e que o favoreçam na nota.

3 Um Sistema de Votação Eletrónica Perfeito, ou Não... (*codenamed*: DONT)

3.1 Breve Introdução

O sistema que tipicamente usamos para votação tem propriedades muito interessantes, aparentemente simples de preencher no mundo físico, mas muito complexas de transportar para o mundo virtual. Exemplos dessas propriedades são: (i) a possibilidade dos votos serem completamente anónimos (por desenho), sem qualquer registo de quem votou em que ou em quem; (ii) a impossibilidade de serem contados votos durante a votação (se as urnas e o procedimento estiverem bem implementados); (iii) a manutenção da garantia de que cada pessoa só vota uma vez; (iv) a garantia de que só votam as pessoas que realmente estão autorizadas para votar (autenticação); e (v) a possibilidade de recontagem de votos sem prejuízo do anonimato.

o objetivo deste trabalho é desenvolver um sistema de votação eletrónica *(e-voting)*, que pode ir do simples sistema que permite a configuração de uma votação e a contagem de votos pelos eleitores, até algo mais elaborado que, por exemplo, apenas permite que uma urna seja aberta quando se juntam simultaneamente vários elementos da comissão eleitoral. No mínimo, o sistema deve ter funcionalidades para dois atores diferentes: um (ou mais) *administrador(es)*, que pode configurar uma votação; e os *eleitores* autorizados a votar. O sistema deve guardar os boletins de votos individuais (para permitir recontagens), mas garantir o anonimato. Os votos devem ser simples de emitir, mas sempre protegidos por autenticação (para assegurar que apenas as pessoas certas podem votar). Uma versão mais elaborada do sistema deve suportar um terceiro tipo de ator (o *membro da comissão eleitoral*). Este terceiro tipo de ator pode ser usado para criar comissões eleitorais, que são os responsáveis por configurar a votação e verificar o resultado final, mas que só podem atuar em grupo (i.e., numa versão avançada do sistema, uma votação é sempre configurada por duas ou mais pessoas, e só essas duas ou mais pessoas é que podem ver os resultados no final). Pretende-se que o requisito da colaboração de entidades seja garantido criptograficamente, usando uma técnica para *partilha segura de segredos* (e.g., *Shamir's Secret Sharing*) para gerar um segredo partilhado por vários utilizadores.

Pensem numa forma de atacar o sistema (uma falha da sua implementação) e dediquem-lhe um pequeno intervalo de tempo na apresentação.

3.2 Funcionalidades Básicas

Em termos de funcionalidades básicas:

- A autenticação dos utilizadores na aplicação ou sistema é feita de uma forma segura (multi-fator ou assinatura digital);
- A aplicação ou sistema permite que utilizadores registados e atribuidos a determinada votação votem, permite guardar os votos e contá-los, mas não permite identificar o autor de determinado voto (e.g., a aplicação gera um código, ligado ao conjunto de eleitores mas não ao eleitor específico, que é atribuido a cada voto);
- A aplicação ou sistema calcula um HMAC-SHA512 de cada voto submetido no sistema com uma chave de integridade gerada aquando da configuração de determinada votação (i.e., gerada para cada votação);
- A aplicação ou sistema guarda todos os boletins de voto na forma cifrada, com a AES-128-CBC, usando uma chave de cifra derivada (com um método seguro) da palavra-passe do administrador ou do segredo dos membros da comissão eleitoral;
- A aplicação ou sistema permite decifrar os boletins de voto, se for necessária a recontagem, voltando a gerar a chave de cifra através do método referido no ponto anterior;
- A aplicação ou sistema deve mostrar o estado da verificação da integridade dos boletins, verificando o código HMAC-SHA512.

3.3 Funcionalidades Avançadas

Em termos de funcionalidades avançadas:

- Para além do código de autenticação de mensagens (HMAC-SHA512), é também incluída uma assinatura digital (implica gerar um par de chaves pública/privada por cada votação configurada);
- Para além de verificar a integridade aquando de uma decifra, é também verificada a assinatura, após ser fornecida a respetiva chave pública à aplicação;
- A aplicação ou sistema suporta a configuração de uma comissão eleitoral e usa um mecanismo de *partilha segura de segredos* (e.g., *Shamirs Secret Sharing*) para gerar os vários segredos que podem reconstruir a chave de cifra e de integridade (por exemplo, se a comissão tiver três elementos, devem ser necessários pelo menos dois comissários para abrir os votos);
- A aplicação permite especificar o número de segredos a gerar e o número mínimo de utilizadores que precisam colaborar para obter a chave de cifra e de integridade;
- Permite-se escolher pelo menos mais uma cifra (diferente da AES) e mais uma função de *hash* para o HMAC;
- Outras funcionalidades relevantes no contexto da segurança do sistema e que o favoreçam na nota.

4 *Shell* Remota Segura (*codenamed*: CARAPAÇA)

4.1 Breve Introdução

Este projeto tem como objetivo a construção de um sistema na arquitetura cliente / servidor que imite o *Secure Shell*. O sistema terá, portanto, uma aplicação servidor e uma aplicação

cliente. Após ser colocada a correr, a aplicação servidor aceita ligações de aplicações cliente, faz a autenticação de utilizadores registados no sistema, estabelece chaves de sessão, e permite que o cliente submeta comandos na máquina anfitriã.

Pensem numa forma de atacar o sistema (uma falha da sua implementação) e dediquem-lhe um pequeno intervalo de tempo na apresentação.

4.2 Funcionalidades Básicas

Em termos de funcionalidades básicas:

- A autenticação dos clientes no sistema é feita de uma forma segura;
- As comunicações entre os clientes e o servidor são protegidas usando algoritmos de criptografia simétrica de qualidade;
- As chaves de cifra (de sessão) são geradas sempre que se inicia nova sessão e são estabelecidas chaves diferentes para todos os tipos de mecanismos criptográficos;
- As mensagens entre clientes e servidor são protegidas com mecanismos de autenticação da origem da informação.

4.3 Funcionalidades Avançadas

Em termos de funcionalidades avançadas:

- O sistema é multi-plataforma (i.e., o mesmo conjunto de aplicações funciona em diferentes sistemas operativos sem muitas modificações/configurações);
- A autenticação é feita usando mecanismos de autenticação forte;
- O sistema suporta autenticação mútua (cliente **e** servidor);
- O sistema suporta vários mecanismos de troca de chaves, nomeadamente mecanismos da criptografia simétrica (e.g., chaves pré-distribuídas ou derivadas de uma palavra-passe) e assimétrica (e.g., Diffie-Hellman ou RSA);
- Em vez de um mecanismo de autenticação da origem da informação, é implementado um mecanismo de assinatura digital;
- Em vez de RSA, são usadas primitivas da criptografia sobre curvas elípticas;
- Outras funcionalidades relevantes no contexto da segurança do sistema e que o favoreçam na nota.

5 Um Distribuidor de Raspadinhas Digitais Seguro (*code-named*: PrivacyEnhancedPrizes)

5.1 Breve Introdução

É comum que os apostadores dos tempos modernos não queiram que seja conhecido o facto de terem ganho um prémio, sobretudo se este for avultado. A ideia princípal deste trabalho é construir um protótipo de um sistema que emule a emissão de prémios aleatoriamente e que garanta a privacidade do utilizador que ganha ou não os prémios. Deve ser

considerada a engenharia e implementação de um sistema com modelo Cliente-Servidor (eventualmente Web ou aplicação mobile), em que o servidor é responsável pela emissão dos prémios. Os vários clientes ligam-se ao servidor para obter uma raspadinha digital que pode ou não conter um prémio. Para efeitos deste trabalho, serão comprometidos alguns pontos, colocando foco noutros. Por exemplo, as raspadinhas são gratuitas, e emitidas a cada utilizador de hora a hora, desde que este esteja ligado. As raspadinhas têm um valor simbólico de 1 (caso ganhe) ou 0 (caso não ganhe nada). Pretende-se que todas as comunicações sejam corretas e confidenciais. Pretende-se que seja utilizado *oblivious transfer* para garantir a privacidade dos utilizadores. Pretende-se que se usem técnicas da criptografia de chave simétrica e pública para garantia da correção e confidencialidade das comunicações.

Pensem numa forma de atacar o sistema (uma falha da sua implementação) e dediquem-lhe um pequeno intervalo de tempo na apresentação.

5.2 Funcionalidades Básicas

Em termos de funcionalidades básicas:

- O sistema integra mecanismos de autenticação forte;
- O registo dos utilizadores é feito de forma segura (considerar, no mínimo, autilização de Diffie-Hellman para troca de uma chave efémera para registo incial dos utilizadores);
- As credenciais de acesso dos utilizadores são guardadas de forma segura;
- As comunicações entre clientes e servidor são cifradas com cifras de qualidade reconhecida;
- As comunicações entre clientes e servidor são protegidas com mecanismos MAC de qualidade reconhecida;
- A distribuição de raspadinhas (e.g., 10 raspadinhas, 1 é ganhadora) dos utilizadores é garantida por um mecanismos de *oblivious transfer* (considerar usar RSA).

5.3 Funcionalidades Avançadas

Em termos de funcionalidades avançadas:

- A autenticação é conseguida através de um protocolo de conhecimento zero;
- O registo dos utilizadores é feito de forma segura e a identidade do servidor é assegurada por certificados digitais;
- Os mecanismos de MAC são substituidos por assinaturas digitais;
- Cada raspadinha ganhadora contém forma de verificação de autenticidade (uma assinatura digital) e formas de anti-repetição;
- Outras funcionalidades relevantes no contexto da segurança do sistema.

6 O Determinador da/o Pagador(a) de Rodadas (*codenamed*: EIPAGADOR)

6.1 Breve Introdução

Um grupo de amigas/os que se juntava todas as sextas-feiras no *Bar do MacLaren* para beber uns copos resolveu desenvolver um sistema que lhes permitisse escolher quem pagava a rodada a cada sexta-feira, mas de forma aleatória (justa) e também de forma a que os obrigasse a estarem juntos por desenho. Assim, o sistema devia recorrer a técnicas que assegurassem que cada um(a) das/os amigas/os tudo faria para que os outros estivessem presentes (por outras palavras, precisavam de *Computação Segura Multipartes*...). Não tendo os conhecimentos necessários para desenvolverem elas/es próprios o sistema, resolveram contratar uma empresa/equipa de elite para lhes fazer o trabalho. Vocs são essa equipa.

O objetivo principal deste trabalho é desenvolver um sistema (eventualmente baseado na web, mas pode ser uma aplicação local para efeitos de protótipo) que aceite o registo de utilizadores (o grupo de $n \in \mathbb{N}$ amigas/os), juntamente com o seu relacionamento, e que, a pedido, escolha aleatoriamente $n - k$ amigas/os para pagarem a próxima rodada, cifrando a lista com uma cifra moderna e segura (e.g., AES-128-CBC) e autenticando-a com um código de autenticação de mensagens (e.g., HMAC-SHA512). A chave para decifrar e verificar deve apenas ser recuperável por um subgrupo de $k \in \mathbb{N}$ amigas/os, configurável no sistema, recorrendo a uma técnica para *partilha segura de segredos* (e.g., *Shamirs Secret Sharing*) para gerar um segredo partilhado por vários utilizadores. E.g., cinco amigos configuram que são precisos quatro amigos para decifrar a lista. Neste caso, a lista deve conter pelo menos dois nomes (por ordem de precedência), uma vez que um dos amigos pode faltar.

As chaves de decifra devem ser enviadas por e-mail para as/os várias/as amigas/os, assim como a lista cifrada e autenticada. Apenas quando se juntam à sexta-feira é que um subgrupo deve conseguir decifrar a lista e descobrir quem paga a rodada.

Pensem numa forma de atacar o sistema (uma falha da sua implementação) e dediquem-lhe um pequeno intervalo de tempo na apresentação.

6.2 Funcionalidades Básicas

Em termos de funcionalidades básicas:

- A aplicação ou sistema integra mecanismos de autenticação forte (assinaturas digitais ou CHAP) para autenticação dos utilizadores;
- A aplicação ou sistema permite a cifra de uma mensagem com AES-128-CBC, gerando automaticamente a chave de cifra;
- A aplicação ou sistema calcula um HMAC-SHA512 com a chave de cifra (que é também chave de integridade) referida no ponto anterior;
- A aplicação ou sistema usa um mecanismo de *partilha segura de segredos* (e.g., *Shamirs Secret Sharing*) para gerar os vários segredos que podem reconstruir a chave de cifra e de integridade. A versão básica gera cinco segredos e requer sempre quatro para reconstruir a chave de cifra;
- É suportada a decifra e a verificação de integridade de criptogramas obtidos de acordo com o que é pedido nos pontos em cima.

6.3 Funcionalidades Avançadas

Em termos de funcionalidades avançadas:

- Para além do código de autenticação de mensagens, é também incluída uma assinatura digital (implica gerar um par de chaves públicas por cada mensagem criada);
- Para além de verificar a integridade aquando de uma decifra, é também verificada a assinatura, após ser fornecida a respetiva chave pública à aplicação;
- A aplicação permite especificar o número de segredos a gerar e o número mínimo de utilizadores que precisam colaborar para obter a chave de cifra e de integridade;
- Permite-se escolher pelo menos mais uma cifra (diferente da AES) e mais uma função de *hash* para o HMAC;
- Outras funcionalidades relevantes no contexto da segurança do sistema e que o favoreçam na nota.

7 Um Testamento Muito Especial (*codenamed*: ItTakes3)

7.1 Breve Introdução

Imagine que uma pessoa decide escrever o seu testamento, deixando-o cifrado (AES-128-CBC) e autenticado por um código de autenticação de mensagens (HMAC-SHA512). Essa pessoa decide dividir a chave de cifra por vários filhos ($n \geq 4$), definindo que é preciso que pelo menos três desses filhos juntem as suas partes para que o testamento seja decifrado. O objetivo principal deste projeto é, portanto, desenvolver uma aplicação ou um sistema de software que permita cifrar, de forma autenticada, um ficheiro, mas cuja chave de cifra deve ser partilhada por vários utilizadores (pelo menos três) e cuja decifra e verificação de integridade requeira, claro, que um certo número desses utilizadores colaborem para autorizar essas operações. Pretende-se que o requisito da colaboração de entidades seja garantido criptograficamente, usando uma técnica para *partilha segura de segredos* (e.g., *Shamirs Secret Sharing*) para gerar um segredo partilhado por vários utilizadores.

Quando o utilizador usa o programa, tem duas hipóteses: cifrar e autenticar ou decifrar e verificar a autenticação. Caso decida cifrar, o programa deixa-o escrever a mensagem a cifrar, o número de partilhas que devem ser gerados e o número mínimo de partilhas necessárias para poder decifrar. A aplicação deve gerar uma chave de cifra e integridade automaticamente, produzir o criptograma, o código de autenticação de mensagens e os vários segredos, guardando-os em ficheiros separados.
Caso decida decifrar, o programa deve aceitar, como *inputs*, os segredos necessários à decifra, o testamento a decifrar e o código de autenticação de mensagens. Deve então devolver o texto-limpo correspondente ou um erro, caso o código de integridade não verifique ou o número de segredos não seja suficiente.

Pensem numa forma de atacar o sistema (uma falha da sua implementação) e dediquem-lhe um pequeno intervalo de tempo na apresentação.

7.2 Funcionalidades Básicas

Em termos de funcionalidades básicas:

- A aplicação ou sistema permite a cifra de uma mensagem com AES-128-CBC, gerando automaticamente a chave de cifra;

- A aplicação ou sistema calcula um HMAC-SHA512 com a chave de cifra (que é também chave de integridade) referida no ponto anterior;
- A aplicação ou sistema usa um mecanismo de *partilha segura de segredos* (e.g., *Shamirs Secret Sharing*) para gerar os vários segredos que podem reconstruir a chave de cifra e de integridade. A versão básica gera quatro segredos e requer sempre três para reconstruir a chave de cifra;
- É suportada a decifra e a verificação de integridade de criptogramas obtidos de acordo com o que é pedido nos pontos em cima.

7.3 Funcionalidades Avançadas

Em termos de funcionalidades avançadas:

- Para além do código de autenticação de mensagens, é também incluída uma assinatura digital (implica gerar um par de chaves públicas por cada mensagem criada);
- Para além de verificar a integridade aquando de uma decifra, é também verificada a assinatura, após ser fornecida a respetiva chave pública à aplicação;
- A aplicação permite especificar o número de segredos a gerar e o número mínimo de utilizadores que precisam colaborar para obter a chave de cifra e de integridade;
- Permite-se escolher pelo menos mais uma cifra (diferente da AES) e mais uma função de *hash* para o HMAC;
- Outras funcionalidades relevantes no contexto da segurança do sistema e que o favoreçam na nota.

8 Estou Cá Pró que Der e Vier (*codenamed*: RightHereWaiting)

8.1 Breve Introdução

O advento do computador quântico poderá ditar o fim de técnicas criptográficas de chave pública muito populares atualamente, nomeadamente a Rivest, Shamir e Adleman (RSA) e esquemas baseados em Curvas Elipticas. Contudo, existem muitas técnicas disponíveis (algumas da criptografia de chave simétrica) que se acreditam serem resistentes à era pós-Quântica.

O objetivo principal deste trabalho consiste em desenvolver uma aplicação ou um sistema de software que permita a proteção de ficheiros, assegurando que essa proteção se mantém mesmo que apareça um computador quântico. As funcionalidades a oferecer são básicas (cifra e decifra, códigos de integridade e assinatura digital), sendo que a maior dificuldade poderá estar no mecanismo a usar para assinaturas digital, que se propõe que seja o esquema de assinatura de Lamport.

Como narrativa, pode considerar que um utilizador quer uma aplicação (que pode ser gráfica ou correr em linha de comandos) que lhe permita criar chaves de cifra ou de assinatura digital (Lamport), cifrar ou decifrar um ficheiro, fazer ou verificar um assinatura, ou fazer ou verificar um código de autenticação de mensagens. Quando abre o programa, este deve listar as opções disponíveis e perguntar o que quer fazer, de entre as várias funcionalidades. Dependendo da funcionalidade pedida, deve continuar a interagir d forma minimamente amigável para produzir os resultados desejados (e.g., caso o utilizador queira

verificar uma assinatura, o programa deve interativamente pedir o nome do ficheiro, a assinatura do ficheiro e a respetiva chave pública). Uma versão mais elaborada o programa pode, por exemplo, suportar vários utilizadores, e guardar as respetivas chaves privadas ou segredos de cifra de forma segura numa base de dados.

Pensem numa forma de atacar o sistema (uma falha da sua implementação) e dediquem-lhe um pequeno intervalo de tempo na apresentação.

8.2 Funcionalidades Básicas

Em termos de funcionalidades básicas:

- A aplicação ou sistema permite gerar chaves públicas e privadas para assinatura digital segundo o esquema de Lamport (para assinaturas digitais);
- A aplicação ou sistema permite calcular a assinatura digital Lamport para o ficheiro indicado e com a chave privada indicada pelo utilizador (efetivamente destruindo a chave privada após ser utilizada). Neste caso, a aplicação devolve ficheiro, a assinatura e a respetiva chave pública de verificação;
- A aplicação ou sistema permite a cifra de um ficheiro com AES-512-CBC, gerando automaticamente a chave de cifra;
- A aplicação ou sistema permite a decifra de um ficheiro com AES-512-CBC, assumindo que lhe é fornecida a respetiva chave de decifra;
- A aplicação ou sistema calcula um HMAC-SHA512 com a chave de cifra (que é também chave de integridade) referida no ponto anterior.

8.3 Funcionalidades Avançadas

Em termos de funcionalidades avançadas:

- A aplicação ou sistema permite verificar um HMAC-SHA512 fornecido junto com determinado ficheiro, assumindo que lhe é dada a respetiva chave de integridade;
- O sistema ou aplicação suporta mais do que um utilizador e integra um mecanismo de autenticação forte ou razoavelmente forte;
- O sistema ou aplicação permite a verificação de assinaturas
- Permite-se escolher pelo menos mais uma cifra (diferente da AES) e mais uma função de *hash* para o HMAC;
- Outras funcionalidades relevantes no contexto da segurança do sistema e que o favoreçam na nota.