Semana 8

Exercício 1)

Tecnologias de *hardware* confiável como os **Trusted Platform Modules** (TPMs) e os **Hardware Security Modules** (HSMs) não são adequadas para dar resposta às exigências de soluções de **Secure Outsourced Computation**, devido a um conjunto de limitações estruturais e funcionais.

Em primeiro lugar, tanto os **TPMs** como os **HSMs** foram implementados com propósitos específicos. Os **TPMs** são essencialmente utilizados para certificar o processo do começo do sistema e garantir a integridade do sistema operativo, enquanto os **HSMs** focam-se em operações criptográficas de elevado desempenho, como a gestão segura de chaves. Estas tecnologias não foram desenhadas para executar código arbitrário ou suportar aplicações complexas que envolvem interações com dados privados. Como consequência, apresentam um desempenho funcional mais limitado e não são capazes de fornecer o ambiente de execução necessário para situações de **Secure Outsourced Computation**.

Para além disso, estas soluções não têm muita capacidade computacional. Dispositivos como **TPMs** ou **smart cards**, por exemplo, não possuem poder de processamento suficiente para executar *workloads* intensivos, que caracterizam ambientes de computação na *cloud*. Esta limitação torna impraticável a sua utilização como solução para cenários onde o cliente pretende executar remotamente uma função num servidor, exigindo ao mesmo tempo garantias sobre a confidencialidade dos dados utilizados e gerados, bem como sobre a integridade do processamento realizado.

Por fim, os **TPMs** e **HSMs** não fornecem mecanismos de isolamento robusto que permitam executar código de forma segura mesmo em sistemas operativos potencialmente comprometidos. Isto é um requisito essencial para garantir que dados sensíveis não possam ser acedidos ou manipulados por atacantes com acesso ao sistema onde se encontram. Pelo contrário, tecnologias como o **Intel SGX** destacam-se precisamente por oferecer esse nível de isolamento.

Exercício 2)

O mecanismo de *attestation* intra-plataforma do **Intel SGX** baseia-se numa chave criptográfica exclusiva por enclave, gerida internamente pelo microprocessador. Neste modelo, quando um enclave A pretende autenticar uma mensagem para um enclave B na mesma máquina, o processador gera um MAC sobre a mensagem e o código de A, utilizando a chave associada ao enclave B. O enclave B, por sua vez, pode recuperar a sua chave localmente e verificar a autenticidade da mensagem. Este processo é eficiente e seguro, mas só funciona dentro da mesma máquina física, onde o processador tem acesso direto às chaves de ambos os enclaves.

No entanto, este mecanismo torna-se insuficiente para *attestation* entre platafor-

mas distintas, ou seja, quando se pretende verificar a integridade e autenticidade de um enclave que está a correr noutra máquina. Este problema deve-se ao facto do processador da máquina local não ter acesso às chaves internas do processador remoto, logo não pode verificar diretamente os MACs gerados.

Para resolver esta limitação, o **Intel SGX** recorre a um mecanismo de *attestation* inter-plataforma, que envolve um novo componente chamado *quoting* enclave, que tem como função validar os dados recebidos e gerar um *quote* assinado. Este enclave tem acesso a uma chave para assinar exclusiva da **Intel**, embutida no *hardware* e inacessível a qualquer outro software. O processo funciona da seguinte forma:

1. O enclave prepara os dados (incluindo um MAC) que serão enviados para o quoting enclave.
2. O *quoting* enclave verifica a integridade da informação recebida e cria o *quote*.
3. O *quoting* enclave assina o *quote* com a **Intel Signing Key**, provando que os dados vêm de uma plataforma SGX legítima.

Assim, o mecanismo de attestation inter-plataforma permite que terceiros verifiquem remotamente a integridade e identidade de enclaves, mesmo sem acesso ao *hardware* interno da plataforma remota, assegurando um elevado grau de confiança na execução segura de código.

Exercício 3)

A principal razão pela qual o **Intel SGX** permite a execução de código geral dentro dos enclaves, enquanto o **ARM TrustZone** restringe-se à execução de aplicações confiáveis no **Secure World**, está relacionada com os diferentes modelos de isolamento e objetivos de *design* subjacentes a cada uma destas tecnologias.

O **Intel SGX** foi concebido com o propósito de viabilizar a execução segura de partes de aplicações potencialmente sensíveis, mesmo em ambientes comprometidos, como sistemas operativos. Para isso, introduz o conceito de enclaves, que são áreas de memória isoladas e protegidas, onde o código pode ser executado com garantias de confidencialidade e integridade.

Pelo contrário, o **ARM TrustZone** adota uma arquitetura dualista, separando o sistema em dois mundos distintos - o **Normal World**, onde corre o sistema operativo convencional, e o **Secure World**, reservado à execução de aplicações consideradas confiáveis (**Trusted Applications**). Neste caso, o ambiente de execução seguro é gerido por um **Trusted OS**, controlado pelo fabricante da máquina ou pela entidade responsável pelo ambiente de execução. Apenas código previamente verificado, assinado e aprovado pode ser executado no **Secure World**, o que significa que não se permite a execução arbitrária de aplicações, para garantir um modelo de segurança estritamente controlado. Esta restrição

no **TrustZone** deve-se a requisitos específicos de segurança. Dado que ambos os mundos partilham o mesmo processador físico, qualquer falha no código do **Secure World** pode comprometer a segurança da aplicação protegida e ainda de todo o sistema. Assim, o *design* do **TrustZone** é uma abordagem mais conservadora e centralizada, onde o fabricante tem controlo sobre o ambiente de execução.

Em suma, o **Intel SGX** oferece um modelo mais flexível e descentralizado, permitindo a execução de código em enclaves isolados, enquanto o **ARM TrustZone** impõe restrições mais rígidas, permitindo apenas a execução de aplicações confiáveis e controladas, com o objetivo de preservar a integridade do ambiente seguro.

Exercício 4)

Dois exemplos de casos onde o **ARM TrustZone** é vantajoso em **comparação** com a arquitetura x86 associada ao **Intel SGX**:

1. O **ARM TrustZone** é amplamente utilizado em **IoT** e dispositivos móveis, como *smartphones* e *smartwatches*, onde a eficiência energética e o baixo consumo de recursos são essenciais. Nestes cenários, o **TrustZone** permite a criação de um ambiente seguro (***Secure World***) para proteger dados sensíveis como biometria, senhas e operações criptográficas, sem a necessidade de *hardware* adicional. Já o **Intel SGX**, por se relacionar a processadores x86, não é tão adequado para estes dispositivos de baixo consumo.
2. O **TrustZone** é particularmente eficaz para garantir o *secure boot*, ou seja, o processo de inicialização segura. Este permite verificar a integridade do sistema operativo antes de o executar. Apesar de o **Intel SGX** oferecer uma forte proteção para partes isoladas de aplicações (os enclaves), ele não protege o sistema operativo desde o momento em que o dispositivo é ligado.

Exercício 5)

Ao desenvolver um serviço de gestão de dados com *hardware* confiável, uma abordagem vantajosa é manter este componente abstrato durante a fase inicial de desenvolvimento. Isto permite uma maior flexibilidade na escolha da tecnologia a ser utilizada, como **Intel SGX** ou **ARM TrustZone**, facilitando eventuais mudanças futuras sem exigir grandes reestruturações do projeto.

Para além disso, ao separar a lógica do "negócio" do ambiente de execução seguro, é possível desenvolver e testar a aplicação mais facilmente. Este ponto também contribui para a flexibilidade do sistema, ao permitir que o mesmo seja executado em diferentes dispositivos ou plataformas.

Por fim, outro benefício importante é a redução da complexidade inicial do

projeto, já que as APIs e requisitos de segurança das tecnologias de *hardware* confiável costumam ser complexos e exigem cuidados específicos.

Exercício 6)

a)

Os enclaves do **Intel SGX** são desenvolvidos para isolar dados sensíveis do restante sistema, inclusive do sistema operativo. Se os dados armazenados dentro de um enclave normal forem revelados inadvertidamente, esta fuga compromete diretamente a confidencialidade e integridade da aplicação. Isto significa que informações sensíveis, como material criptográfico ou dados confidenciais, por exemplo, podem ser acedidas por partes não autorizadas.

Situações em que esta informação possa ser comprometida contradizem completamente o modelo de segurança proposto pelo **SGX**, cuja principal função é precisamente impedir que componentes potencialmente maliciosos do sistema obtenham acesso aos dados protegidos.

b)

O ***quoting* enclave** desempenha um papel importante no processo de **atestação remota** do **Intel SGX**. É este enclave que permite assinar a atestação com uma chave fornecida pela **Intel**. Esta chave é utilizada para provar a autenticidade e a integridade de um enclave perante outras entidades, sejam elas outros enclaves ou não. Se os valores internos deste enclave, especialmente a chave de assinatura, forem expostos ou comprometidos, um atacante conseguiria forjar a atestação remota, permitindo que enclaves maliciosos sejam reconhecidos como legítimos.

Mais uma vez, isto contradiz a metodologia de confiança do **SGX**, ao afetar todas as aplicações e infraestruturas que dependam deste mecanismo de verificação.

c)

Assumir que apenas os dados de *input* e *output* são revelados durante a execução de um enclave, não implica a segurança total dos dados armazenados, devido a ataques *side-channel*. Estes ataques baseiam-se em características da execução, como tempos de resposta, consumo de energia ou emissões electromagnéticas.

Um exemplo concreto é o **ataque de *timing***, em que o atacante mede o tempo que determinadas operações demoram a ser processadas dentro do enclave. Pequenas variações nestes tempos podem revelar informações sobre os dados utilizados, como o valor de *bits* numa chave criptográfica. Outro exemplo são os **ataques à cache**, como o *Prime+Probe* e o *Flush+Reload*, que exploram o facto de a memória em cache ser partilhada entre o enclave e o sistema. Estes ataques permitem inferir informação sensível com base nos padrões de acesso à cache.

Mais concretamente, num caso de estudo, alguns investigadores conseguiram extrair a chave secreta AES em menos de 10 segundos, observando o com-

portamento da cache durante a execução do algoritmo dentro de um enclave SGX. O ataque efetuou medições através de contadores de desempenho (PMCs), disponíveis apenas com privilégios de *root*, exatamente o tipo de atacante contra o qual o SGX propõe proteção.

Posto isto, mesmo que a comunicação com o enclave esteja completamente controlada, a segurança dos dados armazenados internamente continua vulnerável se não forem tomadas medidas adequadas contra este tipo de ataques.