CST0209-Arte de garantir a integridade: Parte 1 - Tipos de controle de integridade de dados

Data do Post	@August 19, 2022 9:00 AM
Status	Not started
Palavras-chave	Arte de garantir a integridade
Fonte	
Pronto	Preparado
IG post	☑
Publicado	☐

Post-Linkedin

A Política de Segurança Cibernética e da Informação é o documento que estabelece conceitos, directrizes e responsabilidades sobre os principais aspectos relacionados à segurança cibernética e segurança da informação, visando preservar os três pilares da segurança cibernética.

Esse artigo, retratará a de forma síntese as políticas e procedimentos da Segurança cibernética. Leia o artigo na íntegra.

Siga-me e veja outros posts no meu Instagram https://www.instagram.com/idaleciosilvatech/

Deixe o like para medirmos a repercussão do conteúdo.
Guarde para rever noutro momento.
Compartilha o conteúdo com os amigos e não só
Comente o que acha do conteúdo e da iniciativa e deixe sugestões.
#cybersecurity #technology #linux #ataquescibernéticos #hacking #crimesvirtuais #despositivosmóveis #mobilidade #protection

Introdução

Nunca os negócios foram tão dependentes da tecnologia. Por esse motivo, pessoas e empresas preocupadas com a sua segurança e dos dados das organizações, preocupação com a consistência dos dados e intendem que ela é uma parte importante para manter a segurança dos dados.

Desse modo, uma das estratégias que pode ser usar para efectivar vontade de minimizar os riscos dos dados, é aplicação de alguns tipos de controle de integridade de dados abordados a seguir.

Algoritmo Hash

Hashing é um processo que garante a integridade dos dados que permite obter, a partir dos dados binários (a mensagem), uma representação de tamanho fixo chamada hash, ou resumo da mensagem. Nesse processo, os dados permanecem inalterados estando em repouso ou trânsito.

O cálculo de um hash recorre ao uso de uma função de dispersão criptográfica para verificar e garantir a integridade dos dados. Também pode verificar a autenticação. As funções de hash são usadas, por exemplo., para substituírem palavras-passe em texto claro, ou chaves de criptografia, porque são funções unidireccionais. Isto significa que, se uma palavra-passe é usada com um algoritmo de hashing específico, o resultado (hash digest) será sempre o mesmo.

Propriedades de um Hash

Hashing é uma função matemática unidireccional que é relativamente fácil de calcular, mas extremamente difícil de reverter.

Para entender o conceito, pensemos num delicioso café

Moer café é uma boa analogia de uma função unidireccional. É fácil moer grãos de café, mas é quase impossível colocar todas as pequenas peças juntas de volta para reconstruir os grãos originais.

Uma função de hash criptográfica tem as seguintes propriedades:

- A entrada pode ser de qualquer comprimento;
- A saída tem um comprimento fixo;
- A função de hash tem um único sentido e não é reversível;
- Dois valores de entrada diferentes raramente resultarão no mesmo valor de hash.

Tipos de Hash

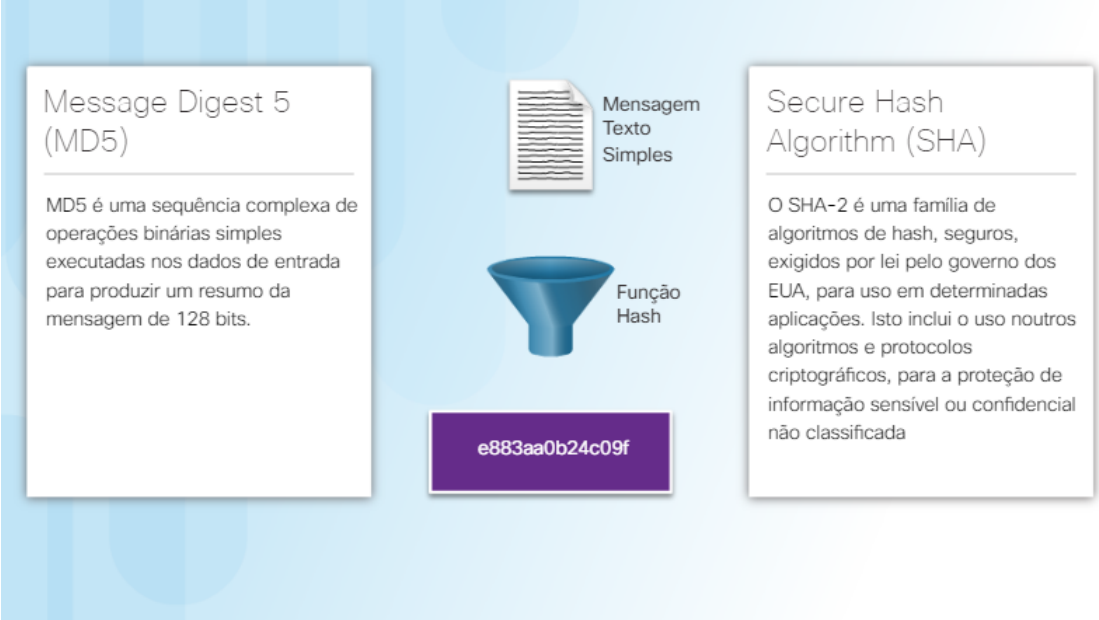
Algoritmo de Hash Simples (Soma de Controlo de 8 bits)

A soma de controlo de 8 bits é um dos primeiros algoritmos de hash, e é a forma mais simples de uma função de hash. Uma soma de controlo de 8 bits calcula o hash começando por converter a mensagem em binário e, a seguir,

organizando a ‘string’ binária em grupos de 8 bits. O algoritmo soma os valores de 8 bits. No passo final é convertido o resultado usando um processo chamado complemento de 2. O complemento do 2 converte um binário para o seu valor reverso e, a seguir, adiciona um. Isto significa que um 0 é convertido em 1, e 1 é convertido em zero. O passo final é somar 1, resultando num valor de hash de 8 bits

Algoritmo do Message Digest 5 (MD5)

Ron Rivest desenvolveu o algoritmo de hash MD5 e, na Internet, ainda há várias aplicações que o utilizam. O MD5 é uma função unidireccional que facilita o cálculo de um hash a partir dos dados de entrada fornecidos, mas que torna muito difícil calcular os dados de entrada usando apenas o valor de hash.



O MD5 produz um valor de hash de 128 bits.

Algoritmo Seguro de Hash (SHA)

O instituto de normas dos, EUA National Institute of Standards and Technology (NIST), desenvolveu o SHA, o algoritmo especificado na norma Secure Hash Standard (SHS).

Seu objetivo é gerar hashes ou códigos exclusivos com base em um padrão com o qual documentos ou dados do computador possam ser protegidos contra qualquer agente externo que deseje modificá-los. Esse algoritmo foi e é um grande avanço no caminho para garantir a privacidade do conteúdo no processamento de informações.

O NIST publicou o SHA-1 em 1994. O SHA-2 substituiu SHA-1, criando uma família de quatro novas funções de hash:

- SHA-224 (224 bits)
- SHA-256 (256 bits)
- SHA-384 (384 bits)
- SHA-512 (512 bits)

O SHA-2 é um algoritmo mais forte, e substitui o MD5. Os SHA-256, SHA-384 e SHA-512 são os algoritmos da próxima geração.

Aplicação dos algoritmos Hash

Utilize funções de hash criptográficas nas seguintes situações:

- Para fornecer prova de autenticidade quando é usado com uma chave de autenticação secreta simétrica, como no protocolo IP Security (IPsec), ou na autenticação de protocolos de encaminhamento;

- Para fornecer autenticação, gerando respostas únicas e unidireccionais aos desafios (challenges) nos protocolos de autenticação;
- Para fornecer prova de verificação de integridade de mensagens, como os usados em contratos assinados digitalmente, e certificados de infra-estrutura de chave pública (PKI), como os aceites no acesso a um site seguro usando um navegador.

Durante a escolha de um algoritmo de hash, utilize o SHA-256 ou superior, pois são actualmente os mais seguros. Evite o SHA-1 e MD5 devido a de falhas de segurança já descobertas nestes algoritmos. Nas redes que se encontram em produção (em uso), implemente o SHA-256 ou superior.

Embora o hash possa detectar alterações acidentais, não consegue proteger contra mudanças deliberadas. Não há nenhuma informação de identificação única do remetente no procedimento de hash. Isto significa que qualquer um pode calcular um hash para quaisquer dados, desde que tenha a função de hash correta. Por exemplo, quando uma mensagem atravessa a rede, um atacante pode interceptar a mensagem, alterá-la, recalcular o hash e anexar o novo hash à mensagem modificada. O dispositivo receptor só irá validar a mensagem recebida contra o hash anexado. Portanto, o processo de hashing é vulnerável a ataques man-in-the-middle (homem-no-meio) e não fornece segurança aos dados transmitidos.

Hash de Ficheiros e de Suporte Digital

A integridade assegura que os dados e a informação estejam completos e inalterados no momento da aquisição. Isto é importante saber quando, por ex., um utilizador descarrega um ficheiro da Internet, ou um investigador forense procura provas num suporte digital.

A área forense digital recorre ao uso de hashing na verificação dos ficheiros constantes num suporte digital. Por exemplo, o investigador cria um hash e uma cópia bit-a-bit do suporte que contém os ficheiros para produzir um clone digital. O investigador compara o hash do suporte digital original com a cópia. Se os dois valores corresponderem, as cópias são idênticas. O facto de o conjunto de bits obtido ser idêntico ao conjunto original de bits, estabelece a fixidez. A fixidez ajuda a responder a várias perguntas:

- O investigador tem os ficheiros que espera?
- Os dados estão corrompidos ou alterados?
- O investigador pode provar que os ficheiros não estão corrompidos?

Agora, o perito forense pode usar a cópia para procurar qualquer prova digital, deixando o original intacto e intocado.

Hash de Palavra-passe

Os algoritmos de hash transformam qualquer quantidade de dados numa impressão ou hash digital, de comprimento fixo. Um criminoso não consegue reverter um hash digital para descobrir a entrada original. Se a entrada for alterada, resulta num hash diferente. Isto funciona para proteger as palavras-passe. Um sistema precisa de armazenar uma palavra-passe num formato que a proteja e ainda conseguir verificar que esta corresponde a um utilizador.

A figura mostra um possível fluxo de eventos para o registo e autenticação de uma conta de utilizador usando um sistema baseado em funções de hash. O sistema nunca grava a palavra passe no disco local, só armazena o hash digital.

Salting

O uso de Sal torna o hashing de palavras-passe mais seguro. Se dois utilizadores tiverem a mesma palavra-passe, eles também terão os mesmos hashes de palavra-passe. Um sal, é um conjunto aleatório de carateres que serve como parâmetro adicional usado no cálculo de um hash de uma palavra-passe. Isto cria um resultado de hash diferente para as duas palavras-passe idênticas, conforme se observa na figura. Uma base de dados armazena tanto o hash como o sal.

Prevenção de Ataques

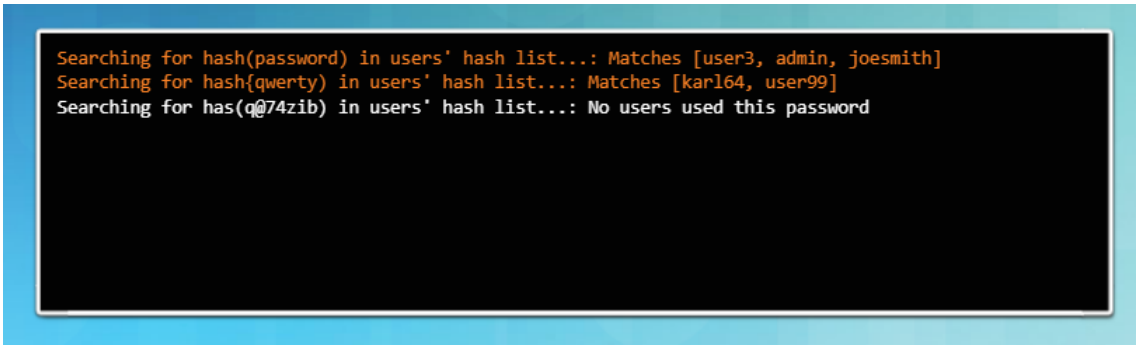
O uso de Sal impede que um atacante use um ataque de dicionário para tentar adivinhar palavras-passe. O uso de Sal também impossibilita usar tabelas de procura e tabelas de arco-íris para quebrar um hash.

Tabelas de Procura

Uma tabela de procura contém os hashes pré-calculados das palavras de um dicionário de palavras-passe com a palavra-passe correspondente. Uma tabela de procura é uma estrutura de dados que permite processar centenas de hash por segundo.

Tabela de Procura Reversa

Este ataque permite que um criminoso inicie um ataque de dicionário ou ataque de força bruta a muitos hashes sem recorrer a uma tabela de procura pré-calculada. O criminoso cria uma tabela de procura onde faz corresponder cada hash constante na base de dados violada, para uma lista de utilizadores. Para cada palavra-passe de suposição, o criminoso determina o seu hash e compara-o com os que constam na tabela de procura criada, para verificar se existem utilizadores cuja palavra-passe corresponda ao palpite do criminoso, como muitos utilizadores têm a mesma palavra-passe, o ataque funciona bem.



Tabelas Arco-íris

As tabelas do arco-íris sacrificam a velocidade de quebra de hashes para tornar as tabelas de procura menores. Uma tabela menor significa que a tabela pode armazenar as soluções para mais hashes na mesma quantidade de espaço.

Implementação de salting

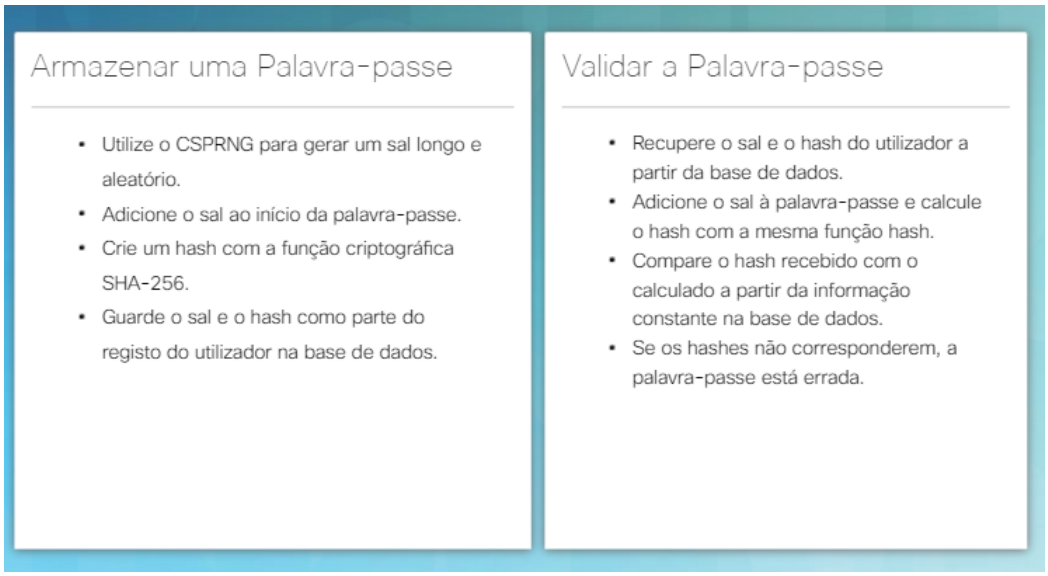
O uso de um Cryptographically Secure Pseudo-Random Number Generator (CSPRNG) é uma boa opção para se gerar o sal. CSPRNGs gera um número aleatório com um alto nível de aleatoriedade e é completamente imprevisível, e, portanto é criptograficamente seguro.

Para implementar o sal com sucesso, siga as seguintes recomendações:

- O sal deve ser único para cada palavra-passe de utilizador.
- Nunca reutilize um sal.
- O cumprimento do sal deve corresponder ao comprimento da saída da função de hash.
- Calcule sempre o hash no servidor, usando uma aplicação 'web'.

O uso da técnica chamada alongamento da chave também ajuda a proteger contra-ataques. O alongamento de chaves torna a função de hash muito lenta. Isto impede que hardware de alto desempenho que pode calcular biliões de hashes por segundo, seja menos eficaz.

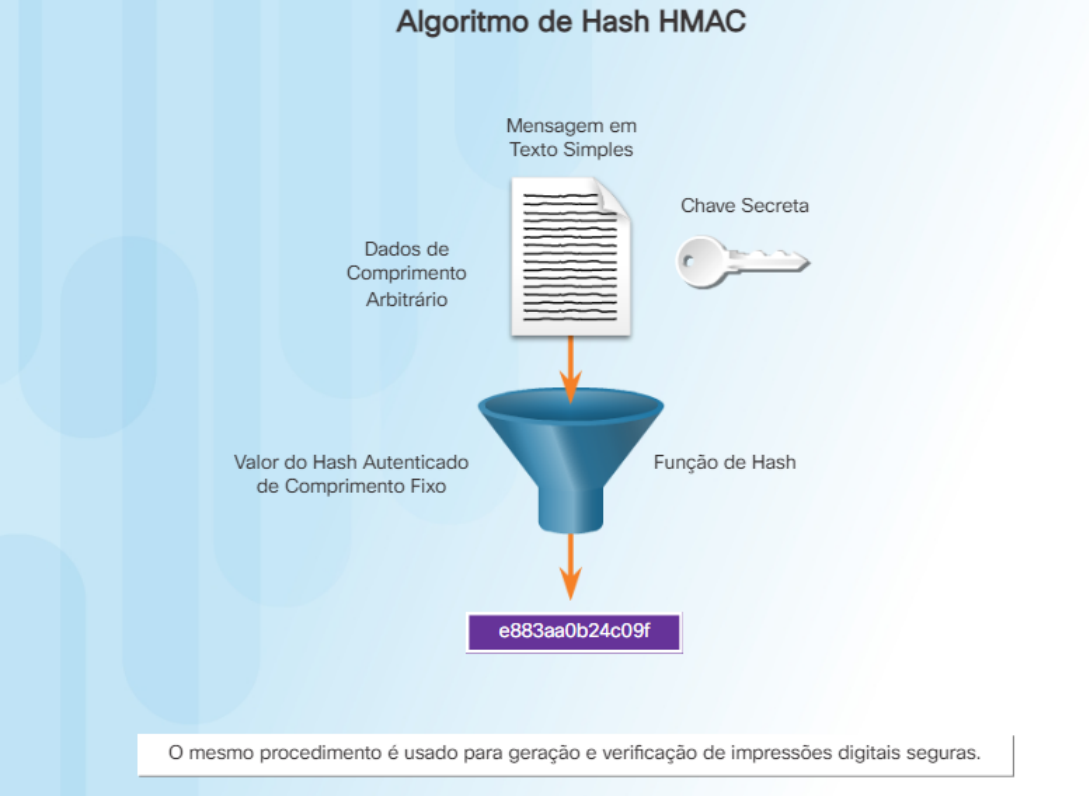
Os passos que uma aplicação da base de dados realiza para armazenar e validar uma palavra-passe salgada são mostrados na figura.



HMAC - código de autenticação de mensagem com chave hash

O próximo passo para impedir que um criminoso lance um ataque de dicionário ou um ataque de força bruta a um hash é adicionar uma chave secreta ao hash. Somente a pessoa que conhece o hash pode validar uma palavra-passe. Uma

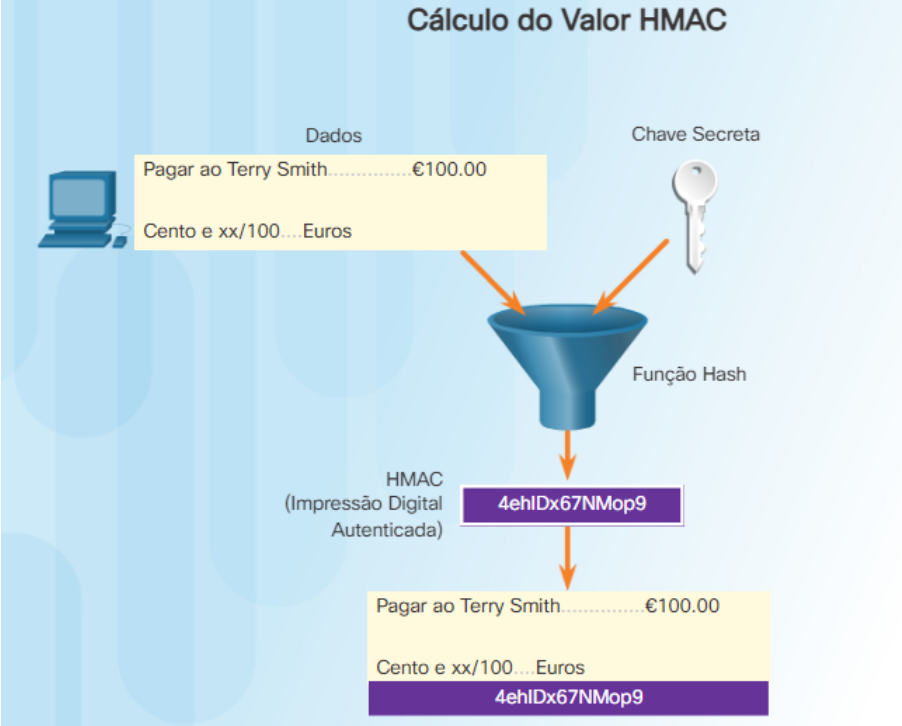
forma de fazer isto é incluir a chave secreta no cálculo do hash, produzindo um keyed-hash message authentication code (HMAC or KHMAC) Os HMACs usam uma chave secreta adicional como entrada para a função de hash. A utilização do HMAC vai mais além do que apenas a garantia de integridade, visto que adiciona autenticação. Um HMAC usa um algoritmo específico que combina uma função de hash criptográfica com uma chave secreta.



Somente o remetente e o receptor conhecem a chave secreta, e a saída da função de hash agora depende dos dados de entrada e da chave secreta. Somente as partes com acesso a essa chave secreta podem calcular o resumo de uma função HMAC. Esta característica trava ataques man-in-the-middle e fornece autenticação da origem dos dados.

Operação do HMAC

Considere um exemplo em que um remetente queira garantir que uma mensagem em trânsito permaneça inalterada e deseja oferecer uma forma para que o receptor autentique a origem da mensagem.



Como se mostra na Figura 1, o dispositivo de envio insere os dados (como o pagamento de Terry Smith de US $100 e a chave secreta) no algoritmo de hash e calcula o resumo HMAC de comprimento fixo ou impressão digital. O receptor obtém a impressão digital autenticada anexada à mensagem.

O dispositivo receptor remove a impressão digital da mensagem e usa a mensagem de texto simples com sua chave secreta como entrada para a mesma função de hash. Se o dispositivo receptor calcular uma impressão digital igual à impressão digital enviada, a mensagem não sofreu alterações. Além disso, o receptor sabe a origem da mensagem porque só ele e o remetente possuem uma cópia da chave secreta partilhada. A função HMAC provou a autenticidade da mensagem.

Aplicação do HMAC

Os HMACs também podem autenticar um utilizador Web. Muitos serviços web usam a autenticação básica, que não cifra o nome de utilizador e a palavra-passe durante a transmissão. Usando o HMAC, o utilizador envia um identificador de chave privada e um HMAC. O servidor procura a chave privada do utilizador e cria um HMAC. O HMAC do utilizador deve corresponder ao calculado pelo servidor.

As VPNs que usam IPsec usam funções HMAC para autenticar a origem de cada pacote e verificar a integridade de dados. Conforme apresentado na figura, os produtos da Cisco usam o hashing para fins da autenticação da entidade, da integridade dos dados, e da autenticidade dos dados:

- Os routers Cisco IOS permitem o uso de hashes com chaves secretas de uma forma como um HMAC para adicionar informação de autenticação nos updates enviados pelos protocolos de encaminhamento.
- Gateways e clientes IPsec usam algoritmos de hash, como o MD5 e o SHA-1 no modo HMAC, para fornecer integridade e autenticidade de pacotes. As imagens do software Cisco, disponíveis em Cisco.com, têm uma soma de controlo baseada emMD5 que permite que clientes possam verificar a integridade das imagens transferidas.

Nota: O termo entidade pode referir-se a dispositivos ou sistemas dentro de uma organização.

Conclusão

Para aumentar a sua segurança e de outras pessoas usar os tipos de controles de integridade de dados apresentados permite maior sucesso nessa luta.

Em suma, as funções hash, o salting e o HMAC são ferramentas essenciais na computação, especialmente quando se trata de excesso de dados. Quando combinados com criptografia, podem ser muito versáteis, oferecendo segurança e autenticação de muitas maneiras.

Siga-me e veja outros posts no meu Instagram https://www.instagram.com/idaleciosilvatech/

Deixe o like para medirmos a repercussão do conteúdo.
Guarde para rever noutro momento.
Compartilha o conteúdo com os amigos e não só
Comente o que acha do conteúdo e da iniciativa e deixe sugestões.
#cybersecurity #technology #linux #ataquescibernéticos #hacking #crimesvirtuais #despositivosmóveis #mobilidade #protection