UNIP Chácara 2

Ciências Da Computação

Método De Criptografia Fernet

APS

N5609C1 - Lucas De Saiz

N5812C8 - Anderson De Sousa Santos

T9701I4 - Ian Dos Santos Firmino

F30FEG0 - Paulo Sergio Satiro Dos Santos

N563DB2 - Ely Jhonathan Maciel Couto

São Paulo 2020

Objetivo

O Objetivo desse trabalho é conseguir aplicar um método de criptografia onde iremos utilizar numa situação em que um navio que foi aprendido pela guarda costeira brasileira por transportar lixo tóxico da Ásia para a região norte do Brasil, nesse trabalho usamos um método de criptografia simétrica onde usaremos Fernet como método

Resumo

A Criptografia é a criptografia é um conjunto de técnicas pensadas para proteger uma informação de modo que apenas emissor e receptor consigam compreendê-la. O protocolo de criptografia pode ser mais ou menos elaborado e técnicas como essas existem desde a antiguidade, com o primeiro sistema de criptografia conhecido tendo surgido no Egito, cerca de 1.900 anos antes de Cristo.

A criptografia tem um apelo especial para assuntos ligados a guerra, mas comerciantes e governantes também podem ver na criptografia uma saída para evitar que pessoas não autorizadas descubram informações sobre suas estratégias, por exemplo. A ideia básica é que este sistema de técnicas cifre uma informação que somente será decifrada por pessoas autorizadas, sem acessos indevidos no caminho.

E entender como a criptografia funciona é simples: o emissor da mensagem utiliza algum protocolo que vai protegê-la, então ela é transmitida até o destinatário, que possui uma chave capaz de “resolver” o problema da criptografia e acessar seu conteúdo.

As chaves podem ser simétricas (quando a mesma chave privada é usada nas duas pontas da transmissão — emissão e recepção) ou assimétricas (quando as chaves de criptografia e descriptografia são distintas, sendo uma pública e a outra privada). Elas são geradas por algoritmos que criam uma sequência de caracteres específica para cada processo. As chaves podem ter tamanhos distintos e, quanto maiores, mais seguras elas se tornam.

As chaves podem ser simétricas (quando a mesma chave privada é usada nas duas pontas da transmissão — emissão e recepção) ou assimétricas (quando as chaves de criptografia e descriptografia são distintas, sendo uma pública e a outra privada). Elas são geradas por algoritmos que criam uma sequência de caracteres específica para cada processo. As chaves podem ter tamanhos distintos e, quanto maiores, mais seguras elas se tornam.

Por que eu deveria usar a criptografia?

Você já entendeu o que é a criptografia e qual a sua finalidade, mas talvez ainda não consiga pensar em um motivo para usar este tipo de proteção, afinal você não é governante nem militar, não tem operações financeiras ou lida com qualquer informação sensível em grande escala. Confira algumas utilidades da criptografia:

1. Proteger informações em trânsito (enviadas por e-mail)

Se por algum motivo você precisa enviar informações sensíveis a seu respeito, como número de cartão de crédito, CPF ou qualquer outro tipo de dado pessoal, optar por criptografá-los antes de fazer isso pode significar um bom incremento de segurança. Assim, o acesso ao conteúdo fica disponível apenas para o destinatário de suas mensagens, que vai possuir a chave para realizar a descriptografia.

2. Proteger dados armazenados nas nuvens

Hoje em dia é muito comum guardar arquivos em serviços de armazenamento na web. Nomes como Dropbox, OneDrive, Google Drive, Box e iCloud são bem recorrentes na vida de muita gente. Em vários momentos, plataformas como estas são mais práticas para guardar e transportar arquivos do que um pendrive. Entretanto, a questão da segurança também é latente aqui. Não é incomum encontrar casos de arquivos que foram vazados a partir de serviços de armazenamento nas nuvens — como o recente vazamento de fotos íntimas do ator Stenio Garcia ou o grande vazamento de fotos íntimas de celebridades de 2014. Em suma, proteger arquivos sensíveis que ficam armazenados neste tipo de lugar também se torna uma questão primordial para evitar maiores problemas. Ou então você pode recorrer a serviços que criptografam dados,
como SpiderOak, TeamDrive, Tresorit e Mega.

3. Proteger arquivos de acesso indevido

Quando o computador ou smartphone é perdido/roubado, você provavelmente começa a esquentar a cabeça pensando que algum desconhecido terá acesso a todos os seus arquivos. Em um gadget portátil com Android, iOS ou Windows Phone, você ainda pode apagar remotamente todos estes dados, mas isso nem sempre é possível em um computador.

Então, o mesmo cuidado que você precisa ter para armazenar na web ou enviar pela internet dados e informações sensíveis deve ser repetido para a proteção de dados armazenados. É lógico que você não precisa bloquear o acesso a todos os arquivos guardados em um disco rígido, mas fazer isso com dados sensíveis pode ser uma saída bem interessante.

Neste caso, o prejuízo de perder o seu computador vai se restringir à peça e ao conteúdo que foram embora, pelo menos ninguém terá acesso aos seus dados. Além disso, proteger dados sigilosos desta forma evita que pessoas não autorizadas (hackers ou não) tenham acesso ao conteúdo guardado em seu PC independentemente de perdas e roubos.

4. Proteger dados de navegação

Quando você acessa uma rede pública de internet (como um Wi-Fi do shopping ou de uma biblioteca, por exemplo), sua navegação pode estar sendo monitorada. Isso pode acontecer inclusive se o seu navegador estiver utilizando um protocolo de segurança (HTTPS). A melhor saída, nesses casos, é recorrer à navegação criptografada.

Isso pode ser alcançado por meio de uma rede virtual privada (VPN) ou ainda utilizar o Tor, pois ambos vão criar "túneis" de conexão criptografada para permitir que você navegue na internet sem ser monitorado.

Técnicas De Criptografia mais utilizadas

1. DES

Data Encryption Standard (DES) é uma das primeiras criptografias utilizadas e é considerada uma proteção básica de poucos bits (cerca de 56). O seu algoritmo é o mais difundido mundialmente e realiza 16 ciclos de codificação para proteger uma informação.
A complexidade e o tamanho das chaves de criptografia são medidos em bits. Quando uma criptografia é feita com 128 bits, significa que 2128 é o número de chaves possíveis para decifrá-la. Atualmente, essa quantidade de bits é considerada segura, mas quanto maior o número, mais elevada será a segurança.

Quando dizemos que um bloco foi criptografado em bits, significa que um conjunto de informações passou pelo mesmo processo da chave, tornando-se ilegível para terceiros.

O DES pode ser decifrado com a técnica de força bruta (o programa testa as possibilidades de chave automaticamente durante horas). Por essa razão, os desenvolvedores precisam buscar alternativas de proteção mais complexas além do DES.

2. 3DES

O *Triple DES* foi originalmente desenvolvido para substituir o DES, já que os hackers aprenderam a superá-lo com relativa facilidade. Houve um tempo em que o 3DES era o padrão recomendado para segurança.
Essa criptografia recebe esse nome pelo fato de trabalhar com três chaves de 56 bits cada, o que gera uma chave com o total de 168 bit. Especialistas no tema argumentam que uma chave de 112 bits é suficiente para proteger os dados.

3. DESX

Essa é outra variante do DES e trata-se de uma solução bastante simples do algoritmo, mas que aumenta exponencialmente a resistência contra ataques de força bruta sem elevar a sua complexidade computacional.

Basicamente, adicionam-se 64 bits antes da encriptação, o que aumenta a proteção de 120 bits contra força bruta. Atualmente, essa tecnologia não é mais imune contra ataques mais sofisticados, como criptoanálises (o programa evolui a cada tentativa de decifração).

4. AES

Advanced Encryption Standard (AES) — ou *Padrão de Criptografia Avançada,* em português — é o algoritmo padrão do governo dos Estados Unidos e de várias outras organizações. Ele é confiável e excepcionalmente eficiente na sua forma em 128 bits, mas também é possível usar chaves e 192 e 256 bits para informações que precisam de proteção maior.
O AES é amplamente considerado imune a todos os ataques, exceto aos ataques de força bruta, que tentam decifrar o código em todas as combinações possíveis em 128, 192 e 256 bits, o que é imensamente difícil na atualidade.

5. Camellia

Desenvolvido em 2000, *Camellia* é uma criptografia que decifra blocos de informações. Trata-se de uma tecnologia com níveis de segurança bastante semelhantes ao AES, já que pode ser processada em 128, 192 e 256 bits.
Camellia pode ser implementada tanto em softwares (programas) quanto hardwares (peças físicas de computador). Também é compatível com tecnologias mais econômicas de 8 bits (smartcards, sistemas de operação em tempo real etc.) até com processadores mais potentes de 32 bits (computadores de mesa).

6. RSA

Rivest-Shamir-Adleman (RSA) foi um dos pioneiros em relação à criptografia de chave pública, seu nome é composto pelos sobrenomes de seus criadores, que também são fundadores da companhia RSA Data Security.
Esse é considerado um dos algoritmos mais seguros do mercado, por essa razão também foi o primeiro a possibilitar a criptografia na assinatura digital.

O RSA funciona da seguinte forma: ele cria duas chaves diferentes, uma pública e outra privada (que deve ser mantida em sigilo). Todas as mensagens podem ser cifradas pela pública, mas somente decifradas pela privada.

Atualmente, essa tecnologia é utilizada em operações rotineiras, como envio de e-mails, compras online, assinatura digital, entre outras atividades.

7. Blowfish

Esse é outro algoritmo desenvolvido para substituir o DES. É uma cifra simétrica que divide as informações em blocos de 64 bits e criptografa cada um deles individualmente.
O *Blowfish* é conhecido por sua velocidade de encriptação e efetividade em geral. Trata-se de uma tecnologia bastante segura, pois há estudiosos no assunto que afirmam que o código não pode ser quebrado.
Ele é completamente grátis, e qualquer indivíduo pode conseguir uma cópia de seu código-fonte, alterar e utilizá-lo em diferentes programas. De forma geral, o *Blowfish* é usado em plataformas de e-commerce para garantir segurança nos pagamentos e proteger senha de acesso dos usuários.

8. Twofish

O *Twofish* é uma variação do Blowfish e também consiste na cifração de blocos simétricos. A diferença é que ele é formado por blocos de 128 bits e chaves de até 256 bits.
A tecnologia é considerada uma das mais rápidas de seu tipo e é ideal para prover segurança de softwares e hardwares. Seu código-fonte também é gratuito, podendo ser manipulado e utilizado por qualquer programador.

Existe outra variação da mesma criptografia chamada *Threefish*, a diferença é que os tamanhos dos blocos são de 256, 512 e 1024 bits, com chaves do mesmo tamanho.

9. SAFER

SAFER (“mais seguro” em português) é uma sigla para *Secure and Fast Encryption Routine*. Consiste na criptografia de blocos em 64 bits, por isso é conhecido como *SAFER SK-64*.
Entretanto, foram encontradas fraquezas nesse código, o que resultou no desenvolvimento de novas versões com diferentes tamanhos de chave, como a SK-40, SK-64 e a SK-128 bits.

10. IDEA

O *Internacional Encryption Algorithm* (IDEA) é uma chave simétrica desenvolvida em 1991, que opera blocos de informações de 64 bits e usa chaves de 128 bits.
O algoritmo utilizado atua de forma diferente, pois usa a confusão e difusão para cifrar o texto. Na prática, ele utiliza três grupos algébricos com operações misturadas, e é dessa forma que o IDEA consegue proteger as informações.

Existem diferentes tipos de criptografia e entendê-los é relevante para que o usuário saiba exatamente como suas informações são protegidas ao utilizar a internet e manusear certificados digitais.
Fonte: Blog da Valid Certificadora.

Nesse Trabalho foi nos dados liberdade para usar qualquer método de criptografia e oque escolhemos foi de Fernet Criptografia simétrica, O Python inclui um pacote que fornece funções de criptografia primitiva com Fernet que garante que uma mensagem criptografada não possa ser manipulada ou lida sem a chave. Fernet é uma implementação de criptografia autenticada simétrica, também conhecida como “chave secreta”. Criptografia simétrica é quando uma chave é usada para criptografar e descriptografar uma mensagem, para que quem quer que seja criptografado possa descriptografá-la. A única maneira de descriptografar a mensagem é saber o que foi usado para criptografá-la, ou seja, como se precisasse de uma senha, uma chave. Para usar criptografia simétrica, usaremos a classe Fernet e a instalação do pacote é feita usando o PIP com um simples comando: **pip install cryptography**

Este é um dos padrões mais seguros, pois o Fernet gera uma chave para que possa haver uma descriptografia segura por isso optamos por ela.

Existem duas maneiras principais de obter uma chave:

1. Você pode gerar uma nova chave ou;
2. Usar uma que foi gerada anteriormente. Ao usar essas chaves para criptografar, mantenha-as em segurança, se você as perder, não poderá descriptografar sua mensagem.
3. A criptografia assimétrica ou criptografia de chaves públicas contorna o grande problema encontrado na criptografia simétrica. Neste método de criptografia é utilizado um par de chaves distintas, uma chave pública e uma chave privada, para promover a codificação e a decodificação. A chave privada deve ser acessível somente para o seu dono. Toda mensagem codificada com a chave privada só pode ser decodificada com a sua respectiva chave pública. E toda mensagem codificada com a

chave pública só pode ser decodificada com o uso de sua chave privada correspondente. Desta forma, se o usuário A deseja enviar uma mensagem codificada para o usuário B, ele pode codificá-la com a chave pública de B, e enviar para B. Somente o usuário B, que possui a chave privada B, conseguirá ler o conteúdo da mensagem enviada por A. A chave pública, como o nome diz, pode ser divulgada a todas as entidades com as quais o dono da chave deseja se relacionar. Os algoritmos de criptografia assimétrica são baseados na resolução de problemas muito complexos que podem envolver curvas elípticas e a fatoração de números primos muito grandes. Devido a este fato, o custo computacional necessário para a codificação e a decodificação de chaves assimétricas é alto. Dentre os algoritmos de criptografia assimétrica mais utilizados, podemos citar o *Rivest Shamir Adleman* (RSA) e o *Digital Signature Algorithm* (DSA).

Código

```
import subprocess
subprocess.run('pip install cryptography', shell=True)
from cryptography.fernet import Fernet

#Bloco 0.0 - Gera a chave unica
priKey = Fernet.generate_key()

#Bloco 0.1 - Escreve a chave unica no arquivo 'key.key'
file = open('.chave\key.key', 'wb')
file.write (priKey)
file.close()

#Bloco 1.0 - Pega a chave preconfigurada no arquivo
file = open('.chave\key.key', 'rb')
key = file.read()
file.close()

#bloco 1.1 - codifica a mensagem
message = input('Digite aqui sua mensagem: ')
encoded = message.encode()

#Bloco 1.2 - Criptografa a mensagem
f = Fernet(key)
encrypted = f.encrypt(encoded)

#Bloco 1.3 -  Mostra a mensagem criptografada
#*Este bloco eh didatico e nao necessariamente estara no programa final
original_message0 = encrypted.decode()
print ("Essa é a mensagem cripitografada: ",original_message0)

#Bloco 2.0 - Le a chave do arquivo pre configurado
file = open('.chave\key.key', 'rb')
key0 = file.read()
file.close()

#Bloco 2.1 - Decriptografa a mensagem
f2 = Fernet (key)
decrypted = decrypted = f2.decrypt(encrypted)

#Bloco 2.2 - Mostra a mensagem
original_message = decrypted.decode()
print ("Essa foi a mensagem enviada: ",original_message)

#Bloco 2.3 - Mostra a chave aplicada.
print ('A chave aplicada foi:',priKey)
```

Bibliografia

https://medium.com/curso-de-programacao-python/criptografia-com-fermet-no-python-7008d0c614bf

https://canaltech.com.br/seguranca/o-que-e-criptografia-e-por-que-voce-deveria-usa-la/

https://cryptoid.com.br/valid/tipos-de-criptografia-conheca-os-10-mais-usados-e-como-funciona-cada-um/

https://www.geeksforgeeks.org/fernet-symmetric-encryption-using-cryptography-module-in-python/

Considerações Finais

A pasta original do trabalho está disponível no link: https://unipead-my.sharepoint.com/:f:/g/personal/lucas_sainz_aluno_unip_br/EseQ3gAJZ7ZFg8o-quhb1mgBHeVwYrybDfpVyd-yQeApJw?e=wL9TA1

Também disponibilizamos nosso repositório no Github: https://github.com/addleking/APSCC2P07.git