PI Criptografia

Cibersegurança e Defesa Cibernética

Grupo

Easy Eats

Entrega relacionada ao projeto ***EASY EATS*** do grupo:

Caique De Oliveira Matos RA: **22023796**

Maria Fernanda Lima dos Santos RA: **22023460**

Pedro Rodante Vicente RA: **22023472**

Thaina Beatriz de Sousa e Silva RA: **22023297**

São Paulo

2023

Dedicatória

Primeiramente, gostaríamos de expressar nossa profunda gratidão ao professor Ronaldo. Suas aulas e seu comprometimento incansável com o ensino foram verdadeiramente inspiradores.

Graças à sua orientação, adquirimos conhecimentos valiosos e ganhamos confiança para explorar novos horizontes na nossa área. Tanto em **Cibersegurança e Defesa Cibernética**, que foi sua matéria do terceiro semestre, quanto em **Fundamentos de Banco de Dados** Que tivemos logo que iniciamos nossos estudos aqui na ***FECAP***.

Seu impacto em nossa jornada acadêmica é importantíssimo, mesmo que isso não seja levantado no dia-a-dia. Muito obrigado, professor Ronaldo, por nos iluminar o caminho para essa nova jornada.

Implementação

Ao longo desse documento, iremos focar em responder às seguintes questões que foram propostas pelo professor Ronaldo e estão expressas abaixo:

PI Criptografia

Envio de documento contendo:

a) Qual parte do projeto seria incluída uma criptografia

b) Qual o algoritmo escolhido para a criptografia

c) Qual a razão pela escolha do algoritmo

d) Se houver, colocar o trecho de código onde será utilizada a criptografia

a) Qual parte do projeto seria incluída uma criptografia?

A criptografia seria incluída em diversas partes do projeto visando garantir a segurança dos dados. Algumas dessas partes incluem:

- **Autenticação das Mesas**: Quando os garçons realizam login das mesas no sistema, suas credenciais devem ser criptografadas para proteger o restaurante de ser acessado em outro local, podendo assim, criar duplicidade e pedidos falsos.
- **Transmissão de Dados**: Quando os dados são enviados entre o cliente e o servidor, eles devem ser criptografados para evitar que sejam interceptados e lidos por terceiros.
- **Armazenamento de Dados**: Os dados sensíveis armazenados no banco de dados, como informações de pagamento dos clientes, devem ser criptografados para proteger contra acesso não autorizado.

b) Qual o algoritmo escolhido para a criptografia?

O algoritmo escolhido para a criptografia seria o ***AES*** *(Advanced Encryption Standard)*.

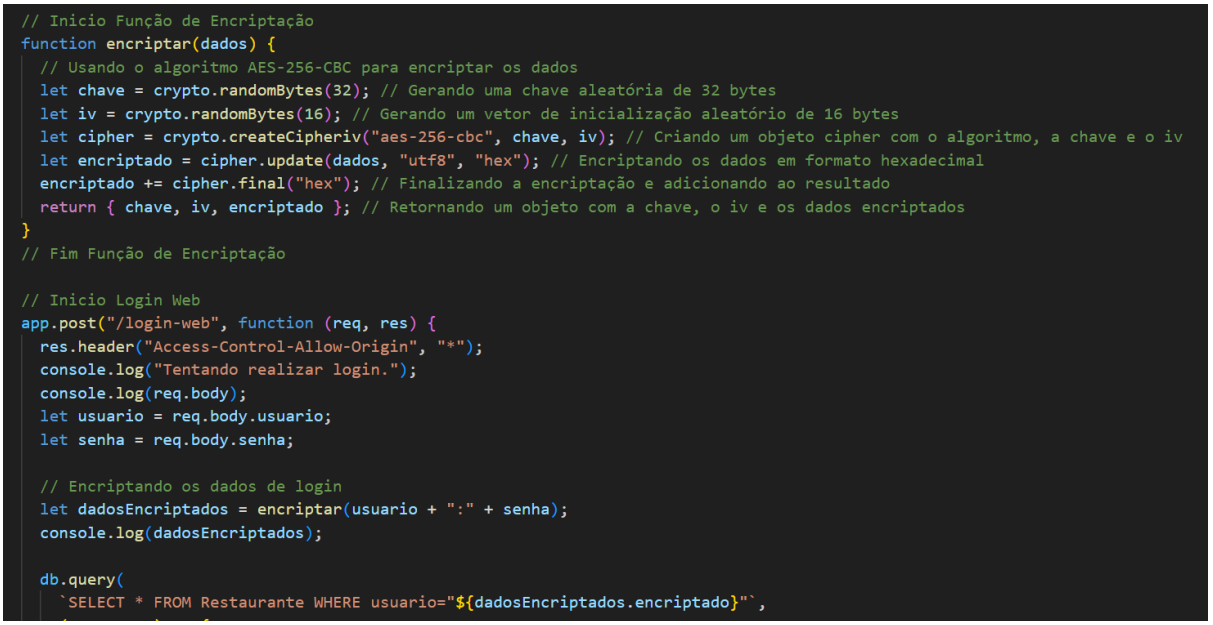
c) Qual a razão pela escolha do algoritmo?

A razão para a escolha do AES é que ele é um padrão de criptografia simétrica estabelecido e amplamente aceito. Ele é rápido, seguro e usado em uma variedade de aplicações, desde a proteção de dados governamentais até a segurança de transações financeiras online.

d) Se houver, colocar o trecho de código onde será utilizada a criptografia?

Abaixo, na próxima página do documento, colocaremos um print que mostra um pouco das implementações de criptografia que estamos estudando e inserindo no projeto. Pensamos em realizar dessa forma, baseado no que aprendemos em aula e nas pesquisas que realizamos com relação às linguagens e frameworks utilizados.

Aqui um exemplo de como a criptografia AES está sendo usada em Node.js no nosso projeto para criptografar os dados:



```
// Inicio Função de Encriptação
function encriptar(dados) {
  // Usando o algoritmo AES-256-CBC para encriptar os dados
  let chave = crypto.randomBytes(32); // Gerando uma chave aleatória de 32 bytes
  let iv = crypto.randomBytes(16); // Gerando um vetor de inicialização aleatório de 16 bytes
  let cipher = crypto.createCipheriv("aes-256-cbc", chave, iv); // Criando um objeto cipher com o algoritmo, a chave e o iv
  let encriptado = cipher.update(dados, "utf8", "hex"); // Encriptando os dados em formato hexadecimal
  encriptado += cipher.final("hex"); // Finalizando a encriptação e adicionando ao resultado
  return { chave, iv, encriptado }; // Retornando um objeto com a chave, o iv e os dados encriptados
}
// Fim Função de Encriptação

// Inicio Login Web
app.post("/login-web", function (req, res) {
  res.header("Access-Control-Allow-Origin", "*");
  console.log("Tentando realizar login.");
  console.log(req.body);
  let usuario = req.body.usuario;
  let senha = req.body.senha;

  // Encriptando os dados de login
  let dadosEncriptados = encriptar(usuario + ":" + senha);
  console.log(dadosEncriptados);

  db.query(
    `SELECT * FROM Restaurante WHERE usuario="${dadosEncriptados.encriptado}"`,
```

```
db.query(
  `SELECT * FROM Restaurante WHERE usuario="${dadosEncriptados.encriptado}"`,
  (err, rows) => {
    console.log(rows);
    if (err) {
      console.log("Erro ao buscar usuário");
      res.send(err);
    } else if (rows.length === 0) {
      console.log("Usuário não encontrado");
      res.send("Usuário não encontrado");
    } else if (rows[0].senha !== senha) {
      console.log("Senha incorreta");
      res.send("Senha incorreta");
    } else {
      if (rows[0].status !== "Ativo") {
        console.log("Status Inativo");
        res.send("Status Inativo");
      } else if (rows[0].ID >= 1) {
        console.log("Bem Vindo Admin");
        console.log(rows);
        res.send(rows);
      } else {
        console.log("Login realizado com sucesso");
        console.log(rows);
        res.send(rows);
      }
    }
  },
```

Aqui usamos o módulo crypto do Node.js para criptografar dados. A função encriptar recebe uma string de dados como entrada e retorna um objeto contendo a chave de criptografia, o vetor de inicialização (IV) e os dados criptografados.

Aqui está o que cada linha do código faz:

- let chave = crypto.randomBytes(32); Esta linha gera uma chave aleatória de 32 bytes.
- let iv = crypto.randomBytes(16); Esta linha gera um vetor de inicialização aleatório de 16 bytes.
- let cipher = crypto.createCipheriv("aes-256-cbc", chave, iv); Esta linha cria um objeto cipher usando o algoritmo AES-256-CBC, a chave e o IV.
- let encriptado = cipher.update(dados, "utf8", "hex"); Esta linha encripta os dados de entrada em formato hexadecimal.
- encriptado += cipher.final("hex"); Esta linha finaliza a encriptação e adiciona ao resultado.
- return { chave, iv, encriptado }; Esta linha retorna um objeto contendo a chave, o IV e os dados encriptados.

Na função encriptar, os dados de login do usuário são encriptados antes de serem enviados ao servidor ou ao banco de dados. A string de dados de login é formada concatenando o nome de usuário e a senha com um dois-pontos (:) entre eles.

Estamos usando o módulo crypto embutido do Node.js para criar uma função de criptografia. Esta função usa o algoritmo AES para criptografar um texto fornecido e retorna o vetor de inicialização (IV) e os dados criptografados como uma string hexadecimal. O IV é necessário para a descriptografia e

é seguro transmiti-lo junto com os dados criptografados, desde que a chave seja mantida em segredo.

Por favor, note que este é apenas um exemplo e a implementação real pode variar dependendo das necessidades específicas do seu projeto. Além disso, é altamente recomendável usar uma biblioteca de criptografia bem testada e manter suas chaves seguras para garantir a segurança dos seus dados.

Conclusão

Ao escolher o módulo crypto do Node.js e implementar o algoritmo AES para criptografia, estamos adotando uma abordagem robusta e confiável para garantir a segurança dos dados. O algoritmo AES é amplamente reconhecido pela sua eficácia e resistência a ataques, proporcionando uma camada sólida de proteção. O uso do vetor de inicialização (IV) acrescenta uma dimensão adicional de segurança, contribuindo para a imprevisibilidade e unicidade do processo de criptografia.

A decisão de transmitir o IV junto com os dados criptografados, mantendo a chave em sigilo, é uma prática segura e eficiente. Isso permite que a informação seja compartilhada de maneira segura, enquanto a chave permanece como a peça central da segurança, mantendo-se fora do alcance de terceiros não autorizados.

Dessa forma, ao adotar esse método de criptografia sólida e bem estabelecida, estamos assegurando a confidencialidade e integridade dos dados, atendendo aos mais altos padrões de segurança na transmissão e armazenamento de informações sensíveis.