Roteiro de Atividade Prática

Nome: _____ Turma: _____

Título da atividade: Implementação de comunicação segura e autenticação em microsserviços para aplicações *mobile*

Contexto:

Com o crescimento das arquiteturas de microsserviços em aplicações *mobile*, garantir a comunicação segura entre serviços, bem como a correta autenticação e autorização, tornou-se essencial. Além disso, monitorar e auditar as atividades dos serviços é fundamental para manter a integridade e a conformidade de segurança. Nesta atividade, você será desafiado a implementar uma solução que garanta a comunicação segura entre microsserviços, configure autenticação e autorização, e implemente um sistema básico de monitoramento e auditoria para acompanhar as operações.

Objetivo:

Guiar o participante na implementação de um sistema que assegure a comunicação entre serviços de forma segura, utilizando *tokens* JWT para autenticação, configurando permissões de acesso entre microsserviços, e monitorando atividades e logs de acesso para auditoria.

Roteiro de Atividade

Passo 1: Configurando o ambiente de microsserviços

1. Escolha uma plataforma para desenvolvimento dos microsserviços (Node.js, Java, ou Python).

2. Crie dois microsserviços básicos: um para Gestão de Usuários e outro para Gestão de Pedidos.
3. Configure um servidor de autenticação (pode ser OAuth 2.0 ou utilizando *tokens* JWT).
4. Defina uma arquitetura de comunicação, utilizando HTTPS, para garantir que os dados trocados entre os serviços sejam criptografados.

Passo 2: Implementação de comunicação segura entre serviços

1. Configuração de HTTPS:
    - Gere um certificado SSL para cada microsserviço e configure-os para usar HTTPS.
    - Verifique a comunicação entre os serviços para garantir que os dados trafeguem de forma criptografada. Exemplo em Node.js:

```
const https = require('https');
const fs = require('fs');

const options = {
  key: fs.readFileSync('server-key.pem'),
  cert: fs.readFileSync('server-cert.pem')
};

https.createServer(options, app).listen(3000, () => {
  console.log('Microserviço rodando em HTTPS na porta 3000');
});
```

Passo 3: Autenticação e autorização entre microsserviços

1. Criação de *tokens* JWT:
    - Configure o servidor de autenticação para gerar um *token* JWT para cada usuário autenticado.

- Cada microsserviço deve validar o *token* antes de processar uma requisição. Exemplo de verificação de *token*:

```
const jwt = require('jsonwebtoken');

app.use((req, res, next) => {
  const token = req.headers['authorization'];
  if (!token) return res.status(403).send('Token é necessário');

  jwt.verify(token, 'secretKey', (err, decoded) => {
    if (err) return res.status(500).send('Falha ao autenticar o token');
    req.userId = decoded.id;
    next();
  });
});
```

2. Definição de *roles* e permissões:
   - Crie *roles* para definir o nível de acesso de cada usuário (como administrador, cliente, gerente).
   - Defina as permissões de cada microsserviço para aceitar apenas requisições de determinados *roles*.

Passo 4: Monitoramento e auditoria das atividades

1. Implementação de logs:
   - Adicione logs de cada requisição feita aos microsserviços, armazenando informações, como horário, IP, tipo de requisição e resposta.
   - Use uma biblioteca de *logging*, como Winston, para Node.js.

2. Configuração de ferramenta de monitoramento:
    - Configure uma ferramenta de monitoramento, como Prometheus ou Elastic Stack, para coletar métricas e *logs* dos microsserviços.
    - Crie um *dashboard* básico que mostre o número de requisições por minuto e possíveis falhas de autenticação.

3. Teste de auditoria:
    - Realize algumas requisições de teste e verifique se os *logs* estão sendo gerados corretamente.
    - Simule tentativas de acesso não autorizado para verificar se o sistema registra os eventos.

Passo 5: Testes finais e validação

1. Realize testes de integração para garantir que a comunicação entre os microsserviços esteja funcionando com segurança.
2. Valide que apenas usuários com *tokens* válidos conseguem acessar os serviços.
3. Revise os *logs* e garanta que eles contenham as informações necessárias para auditoria.

Perguntas para conclusão da atividade:

- Por que é importante utilizar HTTPS para a comunicação entre microsserviços em uma aplicação *mobile*?
- Explique a diferença entre autenticação e autorização em um contexto de microsserviços.
- Como o uso de *tokens* JWT facilita a autenticação entre microsserviços?
- Qual a importância de implementar um sistema de monitoramento e auditoria em uma arquitetura de microsserviços?

Lista de materiais

- Computador com internet;
- Caderno para anotações;
- Uma caneta.