Prof. esp. Thalles Canela

- **Graduado:** Sistemas de Informação - Wyden Facimp
- **Pós-graduado:** Segurança em redes de computadores - Wyden Facimp

- **Professor:** Todo núcleo de T.I. (Graduação e Pós) - Wyden Facimp
- **Diretor:** SCS
- **Gerente de Projetos:** Motoca Systems

Redes sociais:

- **Linkedin:** https://www.linkedin.com/in/thalles-canela/
- **YouTube:** https://www.youtube.com/aXR6CyberSecurity
- **Facebook:** https://www.facebook.com/axr6PenTest
- **Instagram:** https://www.instagram.com/thalles_canela
- **Github:** https://github.com/ThallesCanela
- **Github:** https://github.com/aXR6
- **Twitter:** https://twitter.com/Axr6S

Funcionamento e Segurança

- **Funcionamento da Computação em Nuvem:**
- **Teoria:** Explicação sobre Data Centers, Virtualização e Orquestração.
- **Atividade Prática:** Demonstrar como os provedores de nuvem provisionam recursos usando uma conta de demonstração em um serviço como AWS, Google Cloud ou Azure.

Segurança na Computação em Nuvem

- **Discussão:** Desafios de segurança associados à computação em nuvem.
- **Teoria:**
  - Modelos de Responsabilidade Compartilhada.
  - Criptografia em repouso e em trânsito.
  - Gerenciamento de identidade e acesso (IAM).
- **Atividade Prática:** Configurar políticas básicas de IAM em um serviço de nuvem para proteger recursos.

Desafios de Segurança Associados à Computação em Nuvem:

- **Teoria:** A natureza compartilhada da nuvem introduz preocupações únicas, como a potencial exposição de dados, vulnerabilidades em interfaces e APIs, e questões de conformidade.
- **Exemplo Prático:** Discussão de incidentes de segurança reais na nuvem, como o vazamento de dados do bucket S3 mal configurado.

Modelos de Responsabilidade Compartilhada:

- **Teoria:** Os provedores de nuvem são geralmente responsáveis pela segurança "da" nuvem, enquanto os clientes são responsáveis pela segurança "na" nuvem.
- **Exemplo Prático:** Visualização de um diagrama do AWS, delineando responsabilidades em diferentes serviços (EC2 vs. RDS vs. Lambda).

Criptografia em Repouso e em Trânsito:

- **Teoria:** Proteção de dados quando estão armazenados e enquanto estão sendo transferidos.
- **Exemplo Prático:**
  - Configurar um bucket S3 na AWS para usar a criptografia de lado do servidor (SSE) para proteger dados em repouso.
  - Demonstrar o uso de HTTPS (TLS) para proteger os dados em trânsito para uma aplicação web.

Gerenciamento de Identidade e Acesso (IAM):

- **Teoria:** IAM permite que você controle quem pode fazer o quê em seus recursos de nuvem.
- **Exemplo Prático:**
  - Criar usuários e grupos no IAM da AWS.
  - Atribuir políticas que permitem/denegam acesso a recursos específicos.
  - Demonstração de tentativas de acesso com e sem as permissões adequadas.

Firewalls e Isolamento de Rede:

- **Teoria:** Proteção e segmentação de recursos na nuvem para limitar o acesso indesejado.
- **Exemplo Prático:**
- Configuração de grupos de segurança em uma instância EC2 na AWS.
- Demonstração de tentativas de conexão a uma instância a partir de IPs permitidos e bloqueados.

Monitoramento e Análise de Logs:

- **Teoria:** Identificação de atividades suspeitas ou não autorizadas através do monitoramento contínuo.
- **Exemplo Prático:**
  - Configuração do AWS CloudTrail para monitorar chamadas de API na AWS.
  - Análise de logs para identificar atividades não autorizadas ou suspeitas.

Arquiteturas Fundamentais

- **Modelos de Serviço:** IaaS, PaaS, SaaS
- **Teoria:** Diferenças, vantagens e desvantagens de cada modelo.
- **Atividade em Sala:** Discussão em grupos sobre cenários de uso para cada modelo e apresentação das conclusões.

Arquiteturas de Referência

- **Teoria:**
  - Arquitetura Multicamadas.
  - Microsserviços.
  - Serverless.
- **Atividade Prática:** Desenhar uma arquitetura de referência para um aplicativo de comércio eletrônico usando o conceito de microsserviços.

Arquitetura Multicamadas:

- **Teoria:** Esse modelo divide a aplicação em camadas separadas, com cada camada tendo uma responsabilidade específica. A divisão típica envolve uma camada de apresentação, uma camada lógica de negócio e uma camada de dados.
- **Exemplo Prático:** Um aplicativo de e-commerce.
  - **Camada de Apresentação:** Frontend desenvolvido em Angular mostrando produtos.
  - **Camada de Lógica de Negócio:** Uma API REST em Node.js gerenciando carrinho, pedidos e autenticação.
  - **Camada de Dados:** Banco de dados MongoDB armazenando informações do produto e pedidos do cliente.

Microsserviços:

- **Teoria:** Uma arquitetura de microsserviços divide uma aplicação em serviços menores e independentes que comunicam entre si usando protocolos leves.
- **Exemplo Prático:** Um aplicativo de streaming de música.
  - **Serviço de Usuários:** Gerencia informações e preferências do usuário.
  - **Serviço de Músicas:** Armazena e recupera informações sobre faixas e artistas.
  - **Serviço de Playlists:** Permite aos usuários criar e gerenciar listas de reprodução personalizadas.
  - **Serviço de Recomendação:** Sugere faixas com base no histórico de escuta do usuário.

Arquitetura Orientada a Eventos:

- **Teoria:** Esta arquitetura é baseada na produção, detecção e reação a eventos. É frequentemente usado em sistemas altamente escaláveis e em tempo real.
- **Exemplo Prático:** Um sistema de alerta de inventário.
  - Quando os níveis de estoque de um item caem abaixo de um limite, um "evento" é gerado.
  - Este evento dispara uma série de ações, como reordenar o item ou enviar notificações para gerentes.

Serverless:

- **Teoria:** Serverless permite que os desenvolvedores construam e executem aplicações sem se preocupar com a infraestrutura subjacente. A lógica é executada em resposta a eventos.
- **Exemplo Prático:** Uma API de conversão de imagem usando AWS Lambda.
  - O usuário faz o upload de uma imagem através de um front-end.
  - Um evento é disparado, e o AWS Lambda processa e converte a imagem.
  - A imagem convertida é armazenada em um bucket S3 e uma URL é retornada ao usuário.

Escalabilidade e Alta Disponibilidade

- **Teoria:** Conceitos de escalabilidade horizontal e vertical e estratégias de alta disponibilidade.
- **Atividade em Sala:** Reflexão sobre a importância da escalabilidade e disponibilidade no mundo atual.

Escalabilidade e Alta Disponibilidade

- **Escalabilidade Horizontal:**
- Exemplo: Utilizando Kubernetes para auto-escalonar pods baseado no tráfego.

- **Escalabilidade Vertical:**
- Exemplo: Aumentando as especificações (CPU, memória) de uma instância EC2 na AWS quando se percebe um aumento consistente na carga.

- **Alta Disponibilidade:**
- Exemplo: Usando o AWS RDS (Relational Database Service) com Multi-AZ (múltiplas zonas de disponibilidade) para garantir que, se uma zona falhar, o banco de dados permanecerá online.

Conclusão e Atividade de Casa

- **Resumo:** Recapitulação dos principais pontos da aula.
- **Tarefa de Casa:**
  - Pesquisa sobre um ataque real a um serviço em nuvem e medidas preventivas que poderiam ter sido tomadas.
  - Explorar um serviço PaaS de sua escolha e documentar seus benefícios e desafios.