Emprego DevSecOps no ciclo de vida do software

Você verá o DevSecOps como aplicação de um conjunto de práticas de engenharia de software, processos ágeis e governança de tecnologia da informação considerando o aspecto de segurança desde o início do processo de desenvolvimento de software.

Prof. Bira

Propósito

DevSecOps tem o propósito de garantir o desenvolvimento de produtos de software com o mais alto nível de qualidade e segurança possível, permitindo assim a fluidez de entrega de valor e ritmo de inovação. Velocidade requer organização, segurança e confiança entre as pessoas, para que esse fluxo contínuo de valor entregue seja atingido.

Objetivos

- Reconhecer o DevSecOps e sua influência no ciclo de vida de desenvolvimento de software.
- Reconhecer o processo de orquestração e automação e sua importância para a prática de DevSecOps.
- Reconhecer a modelagem de ameaças, os controles de segurança e a resposta a incidentes.
- Identificar os principais conceitos e padrões de conformidade e governança, bem como a importância da auditoria para a melhoria contínua dos processos das empresas.

Introdução

Com o DevSecOps, você dá um passo importante na jornada de desenvolver aplicativos mais modernos e, acima de tudo, mais seguros. Essa prática de engenharia de software está se tornando cada vez mais popular, especialmente em relação à transformação digital e à segurança da informação. A prática consiste na integração de segurança no ciclo de vida do desenvolvimento de software, desde o início do projeto até a implantação e manutenção do software em produção. Ou seja, DevSecOps garante que a disciplina de segurança seja trabalhada desde a concepção do software a ser desenvolvido, durante sua implementação, até sua manutenção em produção.

Diferente do modelo tradicional de desenvolvimento de software, em que a segurança é considerada apenas na fase final de testes, DevSecOps prega pela integração contínua de requisitos de segurança durante o processo de desenvolvimento, com o objetivo de antecipar, minimizar e até mesmo evitar vulnerabilidades e ameaças à segurança do software em desenvolvimento.

Enquanto DevOps tem como objetivo principal acelerar o processo de desenvolvimento de software por meio de práticas de engenharia de software, automação de processos de desenvolvimento, automação de builds e integração frequente e contínua de pequenos artefatos de software entre outros conceitos, DevSecOps alinha as práticas de desenvolvimento ágil com a segurança da informação, garantindo que a segurança seja parte integrante do processo de desenvolvimento.

Segurança: Modelo tradicional x DevOps

Confira no vídeo a importância da disciplina de segurança em todas as fases do ciclo de vida do desenvolvimento de software.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Ciclo de vida de software

Quando falamos de desenvolvimento de software, a primeira coisa que precisamos considerar é seu ciclo de vida, que normalmente é composto por diversas fases que juntas permitem o desenvolvimento de um produto de software. Não existe um padrão único e que contemple todas as fases possíveis para um ciclo de vida efetivo no desenvolvimento de software, mas podemos destacar fases importantes que devem ser contempladas em qualquer ciclo de vida de desenvolvimento de software. Vamos conferi-las!

Ideação

Fase em que o problema a ser resolvido com um produto de software é explorado e entendido. Nessa fase, os requisitos de negócio que devem ser endereçados pelo produto de software emergem e são adicionados a um backlog de produto.

Design

Fase em que a solução de software para resolver os problemas e requisitos de negócio identificados na fase de ideação é desenhada. Contempla-se arquitetura, definição de componentes e integração entre diferentes serviços (inclusive de terceiros), bem como outras características que devem ser endereçadas pela solução, ou seja, é a fase em que a solução é definida.

Implementação

Fase em que a solução definida na fase de design é implementada, ou seja, construída.

Teste

Fase em que o produto de software desenvolvido é testado quanto a sua aderência de resolução do problema identificado na fase de ideação.

Implantação

Fase em que o produto de software desenvolvido e testado é posto em produção para seu efetivo uso por parte dos usuários.

Evolução
Fase em que o produto de software é evoluído, baseado no feedback de seus usuários.

A descrição das fases não deixa claro em que momento a disciplina de segurança deve ser contemplada. Na maioria das vezes, a subjetividade levava à negligência dessa disciplina importante por parte das equipes de desenvolvimento de software, que só se preocupavam com segurança quando algum problema era relatado pelos usuários.

Com o advento de **DevSecOps**, a segurança passa a ser uma disciplina pensada e tratada em todas as fases do ciclo de vida de desenvolvimento, desde a ideação até a sua evolução, sendo parte integrante do dia a dia das pessoas envolvidas em seu desenvolvimento.

Segurança no modelo tradicional de desenvolvimento de software

No modelo de desenvolvimento de software tradicional, a disciplina de segurança é normalmente endereçada como um aspecto isolado do processo de desenvolvimento, sendo muita das vezes uma fase adicional ao ciclo de vida, ou até mesmo uma atividade na fase de testes desses ciclos tradicionais. Isso porque a disciplina de segurança é identificada como um requisito adicional aos requisitos funcionais das soluções de software, permitindo que os times de implementação enxerguem segurança como uma fase posterior ao desenvolvimento das funcionalidades de negócio.

> Os times focam na implementação de artefatos de software que resolvem os problemas de negócio, e só depois se preocupam em testar esses artefatos quanto aos aspectos de segurança necessários.

Muitas vezes os testes de segurança somente são realizados na fase diretamente anterior a sua implantação. Normalmente os testes são realizados por uma equipe especializada em segurança que utiliza ferramentas de terceiros para realizar diversos tipos de testes, como testes de penetração, com o objetivo de identificar falhas e vulnerabilidades no sistema desenvolvido.

O problema é que essa abordagem é insuficiente e ineficiente, pois as vulnerabilidades somente serão descobertas tardiamente no processo de desenvolvimento, gerando um conjunto de bugs críticos de segurança que devem ser endereçados prioritariamente sobre novas funcionalidades, o que pode levar a atrasos no lançamento do software e a possíveis violações de segurança.

Além disso, a disciplina de segurança não é considerada uma responsabilidade compartilhada de todos os envolvidos no processo de desenvolvimento, levando a uma falta de conscientização e compreensão sobre a importância da segurança por todos os envolvidos no desenvolvimento do produto de software.

Segurança no modelo DevSecOps de desenvolvimento de software

No modelo de desenvolvimento DevSecOps existe uma abordagem completamente diferente da disciplina segurança. Com DevSecOps a disciplina segurança é considerada parte integrante do processo de desenvolvimento do software e permeia por todas as fases de seu ciclo de desenvolvimento, desde a ideação até sua operação e evolução.

Essa mudança de paradigma coloca a disciplina de segurança como um dos pilares centrais do desenvolvimento do produto de software, fazendo com que os aspectos de segurança sejam considerados em todos os momentos.

Uma das principais maneiras pelas quais o DevSecOps aborda a segurança é por meio da utilização de técnicas e ferramentas de automação. Ferramentas de automação são usadas como base para que continuamente possamos avaliar a segurança do código que está sendo desenvolvido.

Essas ferramentas também servem para detectar e corrigir vulnerabilidades de segurança à medida que são encontradas. Elas incluem:

- Análise estática de código
- Testes de penetração automatizados
- Monitoramento de vulnerabilidades em tempo real

Outra maneira pela qual o DevSecOps endereça a segurança é por meio do envolvimento de equipes de segurança em todo o processo de desenvolvimento. Isso significa que os especialistas em segurança são incorporados às equipes de desenvolvimento e operações desde o início, em vez de serem chamados apenas no final do processo para realizar uma revisão de segurança. Essa abordagem colaborativa ajuda a identificar e resolver problemas de segurança mais cedo no processo de desenvolvimento, reduzindo o tempo e o custo associados à correção de problemas de segurança no final do ciclo de vida do software.

Além disso, a abordagem DevSecOps promove a cultura de segurança em toda a organização, permitindo que todos os membros da equipe sejam responsáveis pela segurança do software, não apenas os especialistas em segurança. Isso é alcançado a partir da conscientização e do treinamento de segurança para todos os membros da equipe, bem como por meio da implementação de políticas e práticas de segurança consistentes em toda a organização.

Resumindo

O DevSecOps aborda a segurança de forma mais integrada e colaborativa do que os modelos tradicionais de desenvolvimento de software. A segurança é incorporada em todas as fases do ciclo de vida do software, é automatizada sempre que possível e é abordada como uma responsabilidade de toda a equipe. Essa abordagem ajuda a garantir que o software desenvolvido seja mais seguro e confiável, reduzindo o risco de violações de segurança e minimizando os custos associados à correção de problemas de segurança no final do processo de desenvolvimento.

Propósito e benefícios do DevSecOps

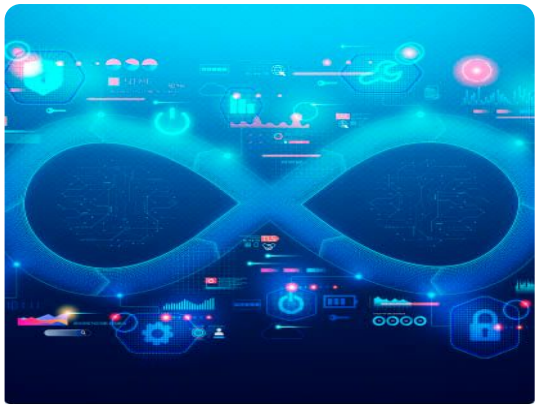
Entenda no vídeo o propósito e os benefícios do DevSecOps, uma prática que integra segurança em todo o ciclo de vida do desenvolvimento de software.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Propósito de DevSecOps

O propósito é garantir que a disciplina de segurança seja integrada ao longo de todo o ciclo de vida de desenvolvimento do software, desde sua conceção até a sua operação e evolução. Mas não se restringe apenas aos aspectos técnicos utilizados no desenvolvimento dos produtos de software, DevSecOps vai além, criando uma cultura em que a segurança seja integrada em todos os aspectos e atividades das empresas. Assim, não se restringe ao aspecto técnico e se torna uma preocupação de todas as áreas e pessoas dentro das empresas.



O DevSecOps também ajuda a promover a cultura de segurança dentro das organizações de desenvolvimento de software, tornando a segurança uma responsabilidade compartilhada por todos os membros da equipe. Isso pode levar a uma maior conscientização e responsabilidade em relação à segurança do software e dos dados do usuário.

Benefícios de DevSecOps

É notório que quando adotamos um modelo de desenvolvimento de produtos de software baseado em uma cultura e técnicas de DevSecOps vamos colher diversos benefícios. Vamos destacar alguns!

- **Redução de custos**: Ao se tentar antecipar e minimizar possíveis vulnerabilidades e ameaças à segurança do software, a prática de DevSecOps tende a reduzir custos com ações corretivas e de reparação de danos, que muitas das vezes podem ser intangíveis.
- **Melhoria da qualidade do software**: Pensar e integrar segurança desde o início do processo de desenvolvimento e mantendo durante todo o ciclo de vida de um software, garante a melhor qualidade do software possível, fazendo com que o software atenda aos requisitos de segurança, performance e disponibilidade.
- **Maior eficiência do processo**: A automação de testes e a integração contínua permitem uma entrega mais rápida e confiável de novas funcionalidades e correções de segurança.
- **Maior transparência**: A integração contínua de segurança no processo de desenvolvimento pode aumentar a transparência do projeto, permitindo que os desenvolvedores entendam melhor os riscos de segurança e tomem medidas preventivas.
- **Maior conformidade regulatória**: O DevSecOps incorpora a conformidade com padrões e regulamentos de segurança, ajudando as organizações a cumprir os requisitos de conformidade e a evitar penalidades e multas.
- **Melhor resiliência**: A adoção de práticas DevSecOps pode melhorar a resiliência do sistema, permitindo que a organização se recupere mais rapidamente de possíveis violações de segurança ou interrupções no sistema.
- **Maior visibilidade e transparência**: A integração contínua de segurança no processo de desenvolvimento pode aumentar a transparência do projeto, permitindo que os desenvolvedores entendam melhor os riscos de segurança e tomem medidas preventivas.
- **Aceleração da entrega de software**: Ao integrar segurança no processo de desenvolvimento, as equipes podem antecipar e resolver problemas de segurança mais rapidamente, o que pode acelerar a entrega de software.
- **Melhor gerenciamento de riscos**: O DevSecOps permite uma abordagem proativa na gestão de riscos de segurança, ajudando as organizações a identificar e gerenciar melhor possíveis ameaças.
- **Maior agilidade**: A incorporação de práticas DevSecOps pode melhorar a agilidade das organizações, permitindo que elas respondam rapidamente às mudanças no ambiente de segurança e às demandas do mercado.
- **Melhor reputação e confiança**: Uma abordagem proativa em relação à segurança do software pode ajudar a construir uma reputação positiva e a confiança do cliente, aumentando a lealdade e a fidelidade do cliente.

Em resumo, DevSecOps é uma prática que visa integrar segurança no ciclo de vida do desenvolvimento de software, com o objetivo de antecipar e minimizar possíveis vulnerabilidades e ameaças à segurança do software. A prática de integrar segurança desde o início do processo de desenvolvimento pode ajudar a reduzir custos, melhorar a qualidade do software, aumentar a transparência e acelerar a entrega do software.

Verificando o aprendizado

Questão 1

Assinale a alternativa que apresenta como é reconhecida a disciplina de segurança no modelo tradicional de desenvolvimento de software.

A

É identificada como um requisito adicional aos requisitos funcionais das soluções de software, fazendo os times de implementação enxergarem a disciplina de segurança como uma fase posterior ao desenvolvimento das funcionalidades de negócio, sendo uma preocupação apenas nas fases de testes.

B

É identificada como um dos requisitos de negócio das soluções de software, fazendo o desenvolvimento das funcionalidades ser iniciado somente depois do mapeamento de todas as possíveis falhas de segurança que a solução pode vir a ter.

C

É identificada como um requisito adicional aos requisitos funcionais das soluções de software, fazendo os times de implementação enxergarem a disciplina de segurança como uma fase inicial ao desenvolvimento das funcionalidades de negócio, sendo uma preocupação constante apenas nas fases de testes.

D

É identificada como um dos requisitos não funcionais das soluções de software, fazendo o desenvolvimento das funcionalidades ser iniciado somente depois do mapeamento de todas as possíveis falhas de segurança que a solução pode vir a ter.

E

É identificada como um requisito adicional aos requisitos funcionais das soluções de software, fazendo os times de implementação enxergarem a disciplina de segurança como uma paralela ao desenvolvimento das funcionalidades de negócio, sendo uma preocupação dos times de teste.

A alternativa A está correta.

No modelo tradicional de desenvolvimento de software, a disciplina de segurança é geralmente vista como um requisito adicional aos requisitos funcionais, e muitas vezes é considerada apenas uma fase posterior ao desenvolvimento das funcionalidades de negócio. Isso significa que a segurança é tratada como uma preocupação apenas nas fases de testes, e não é uma preocupação constante ao longo do processo de desenvolvimento. Esse modelo pode levar a problemas graves de segurança, já que as falhas de segurança podem ser descobertas tardiamente no processo de desenvolvimento, tornando mais difícil e caro corrigi-las.

Questão 2

Assinale a alternativa de como é reconhecida a disciplina de segurança no modelo DevSecOps de desenvolvimento de software.

A

É considerada uma fase do processo de desenvolvimento do software que pode ocorrer no início ou no final do ciclo de desenvolvimento.

B

É considerada uma subfase do processo de desenvolvimento do software, sendo vista como parte do ciclo de desenvolvimento, desde a ideação até sua operação e evolução.

C

É considerada parte integrante do processo de desenvolvimento do software e permeia por todas as fases de seu ciclo de desenvolvimento, desde a ideação até sua operação e evolução.

D

É considerada parte integrante da esteira de desenvolvimento do software, sendo executada apenas por aplicações específicas de segurança.

E

É considerada uma disciplina importante que tem seu desenvolvimento em paralelo ao desenvolvimento do produto de software.

A alternativa C está correta.

O modelo DevSecOps tem como um de seus principais pilares a integração da segurança ao longo de todo o ciclo de vida do desenvolvimento de software. A segurança é uma disciplina que deve estar presente em todas as etapas do ciclo de vida do software, desde o planejamento até a implantação e manutenção. A ideia é que a segurança seja incorporada desde o início do processo de desenvolvimento, em vez de ser tratada como uma fase posterior. Isso permite que as equipes identifiquem e corrijam problemas de segurança o mais cedo possível, minimizando o risco de falhas de segurança no produto final.

Processo de automação e orquestração

Confira no vídeo a importância dos conceitos de automação e orquestração na área de DevOps e DevSecOps.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

A importância dos conceitos de automação e orquestração

Tanto DevOps quanto DevSecOps têm como um de seus pilares a aplicação dos conceitos de automação e orquestração das atividades necessárias no desenvolvimento das soluções de software. O objetivo principal da aplicação desses conceitos é garantir que atividades rotineiras, que muitas das vezes seriam executadas por pessoas, sejam executadas pelos computadores.

Exemplo

Imagine que, há alguns anos, toda vez que uma nova versão de um software fosse lançada, os profissionais precisassem testar manualmente todas as funcionalidades, tanto as novas quanto as antigas. Isso era necessário para garantir que o novo desenvolvimento não afetasse as funcionalidades existentes. Nesse cenário, as empresas costumavam ter áreas com muitas pessoas responsáveis por realizar esses testes repetitivos. Esse processo era demorado, propenso a erros - afinal, dependia do trabalho humano - e extremamente ineficiente.

A indústria de desenvolvimento de produtos de software precisava encontrar alternativas para melhorar esse cenário, tornando-o mais seguro, rápido e eficiente. E a solução para isso foram os conceitos de automação e orquestração.

Automação

É fundamental para DevSecOps porque permite que tarefas repetitivas sejam efetuadas de maneira automatizada, como: realização de testes funcionais e de segurança, compilação de código, publicação de artefatos em ambientes, verificação de vulnerabilidades, provisionamento de recursos etc. A automação é importante, pois aumenta a eficiência economizando tempo e recursos, além de reduzir erros humanos.

Justamente por automatizar tarefas repetitivas e garantir que sejam executadas de maneira muito mais rápida e segura, a automação permite que novas versões de uma solução de software sejam entregues de maneira mais rápida e contínua, o que é essencial para acompanhar as demandas do mercado.

Orquestração

É o conceito que se refere à capacidade de coordenar diferentes ferramentas e processos de automação para criar uma cadeia de eventos contínua.

A orquestração é responsável por executar as tarefas programadas na sequência correta, definir dependências e gerenciar a execução em vários ambientes, garantindo que todo o processo de entrega de artefatos de software esteja integrado e funcionando de maneira contínua e eficiente.

DevSecOps tem a orquestração como um de seus principais pilares, pois garante que as etapas necessárias para se garantir os critérios de segurança de um artefato de software serão executadas de maneira adequada e que o processo de análise seguirá um caminho predeterminado de acordo com o resultado das tarefas executadas.

Benefícios

Como principais benefícios da automação e orquestração, podemos destacar:

Eficiência

Ao aplicar técnicas e ferramentas de automação e orquestração, diversas tarefas repetitivas podem ser executadas de maneira muito mais rápida e com menor propensão a erros humanos, garantindo maior eficiência no processo de desenvolvimento de produtos de software. Além de permitir que os profissionais se concentrem em outras atividades que permitam a aceleração do desenvolvimento das entregas de funcionalidades e novos recursos.

Mitigação de erros

Por se tratar de processos automatizados e orquestrados, removendo o fator humano das atividades repetitivas, mitiga-se a possibilidade de erros eventuais causados pela má execução de uma pessoa.

Visibilidade

A orquestração fornece uma visão geral de todo o processo de desenvolvimento do produto de software, permitindo que as equipes identifiquem rapidamente possíveis problemas e tomem as ações de correção de maneira mais acelerada, garantindo assim o monitoramento de todo o pipeline do ciclo de entrega dos artefatos de software.

Redução de custos

A automação e a orquestração ajudam a reduzir os custos de desenvolvimento dos produtos de software, permitindo que as equipes realizem tarefas de maneira mais eficiente.

Confiabilidade

Aumento da confiabilidade do pipeline de entrega, garantindo que todas as etapas estejam completas antes que a próxima seja iniciada.

Segurança

A automação e a orquestração ajudam a aumentar a segurança do processo de entrega, detectando e corrigindo vulnerabilidades mais rapidamente.

CI/CD e DevSecOps

Acompanhe no vídeo como as práticas de continuous integration (CI) e continuous delivery (CD) são fundamentais para o sucesso de abordagens DevOps e DevSecOps.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Continuous integration (integração contínua) e continuous delivery (entrega contínua) são práticas de engenharia de software oriundas dos conceitos de automação e orquestração, que são pilares fundamentais para as abordagens DevOps e DevSecOps. As práticas e técnicas são tão essenciais que pode-se dizer que sem elas, DevOps e DevSecOps não seriam viáveis, devido aos benefícios que elas proporcionam como suporte para essas abordagens.

Continuous integration (CI)

É uma prática de engenharia de software que tem como objetivos:

- Automatizar o processo de compilação, teste e implantação de código-fonte, de maneira mais rápida, eficiente e frequente possível. Essa prática foi popularizada por Martin Fowler e Kent Beck em seu livro Continuous Integration: Improving Software Quality and Reducing Risk.
- Detectar erros o mais cedo possível no ciclo de vida do desenvolvimento de software, a partir da frequente integração da base de código-fonte pelos desenvolvedores, antes que se tornem problemas maiores e mais difíceis de corrigir. Isso é feito por meio da execução automatizada de diversas fases, desde a análise estática de código e de testes em busca de violações de estilo e outras anomalias.

A CI também ajuda a facilitar a colaboração entre os membros da equipe de desenvolvimento, garantindo que o código-fonte seja frequentemente integrado ao repositório principal, permitindo que todos os desenvolvedores trabalhem a partir de uma base atualizada frequentemente.

Para implementar a CI, é necessário o uso de ferramentas que permitam a automação do processo de compilação e teste do código-fonte, como Jenkins, GitLab CI/CD, Azure DevOps, entre outras.

Além disso, é necessário que a equipe de desenvolvimento adote uma cultura de colaboração e comunicação constante para que a prática de integração contínua seja bem-sucedida, preferencialmente alinhada a uma cultura ágil.

Continuous delivery (CD)

É uma prática de engenharia de software que visa entregar alterações de código de forma frequente, segura e confiável. O objetivo da CD é alcançado por meio da automação do processo de entrega dos artefatos de software, incluindo testes e implantação automatizada.

A CD ligeiramente difere da CI, pois se concentra na entrega confiável e automatizada de software em um ambiente de produção. No entanto, a CI é um componente fundamental da CD, já que permite aos desenvolvedores integrar suas alterações de código com frequência e detectar erros mais rapidamente, sendo assim duas práticas complementares.

CI/CD e DevSecOps

DevSecOps acrescenta uma dimensão de segurança à prática de CI/CD. Enquanto a integração contínua e a entrega contínua se concentram principalmente em automatizar e agilizar o processo de desenvolvimento e entrega de software, DevSecOps enfatiza a importância de incorporar práticas de segurança em todas as etapas do ciclo de vida do desenvolvimento de software.

Assim, o DevSecOps adiciona controles de segurança automatizados em todas as etapas dos pipelines de CI/CD. Confira tais práticas de segurança!

- Análise estática de código-fonte
- Verificação de vulnerabilidades
- Análise de configuração de segurança
- Testes de penetração automatizados

Essas práticas de segurança ajudam a identificar e corrigir vulnerabilidades em um estágio inicial do desenvolvimento, garantindo que as ameaças sejam mitigadas antes de chegarem à produção.

Além disso, o DevSecOps incentiva a colaboração entre equipes de segurança, desenvolvimento e operações, para garantir que os requisitos de segurança sejam considerados desde o início do processo de desenvolvimento de software.

A abordagem DevSecOps também promove a implementação de controles de segurança de forma contínua e o monitoramento em tempo real de ameaças e vulnerabilidades, permitindo que as organizações respondam rapidamente a incidentes de segurança e reduzam o risco geral de segurança de seus sistemas de software.

Scripts e estruturas de testes

Entenda no vídeo como scripts e estruturas de testes são práticas fundamentais para DevSecOps.

> Conteúdo interativo
>
> Acesse a versão digital para assistir ao vídeo.

Scripts e estruturas de teste são fundamentais para o DevSecOps, pois são as ferramentas usadas na integração e entrega contínua para avaliar os parâmetros de segurança. Eles desempenham um papel crucial na garantia da qualidade e segurança do software em desenvolvimento. Vamos conhecê-las!

Scripts

São arquivos que contêm uma sequência de comandos e instruções que podem ser interpretados e executados por um programa ou sistema operacional. Esses comandos e instruções são escritos em uma linguagem de script, que é uma linguagem de programação projetada para ser facilmente legível por humanos e executável por computadores.

Os scripts são usados para automatizar tarefas e processos repetitivos em sistemas de computador, como instalação de software, configuração do sistema, gerenciamento de arquivos e pastas, entre outros. No contexto de DevSecOps, os scripts são comumente usados para realizar testes de segurança automatizados e para integrar as ferramentas de segurança ao pipeline de entrega contínua.

> Scripts são importantes no contexto de DevSecOps porque permitem automatizar tarefas repetitivas e complexas, melhorando a eficiência do processo de desenvolvimento e segurança do produto de software.

Com o aumento da complexidade dos sistemas de software, há uma crescente necessidade de testes automatizados para garantir que os sistemas estejam em conformidade com os requisitos de segurança. Dentre as funções que os scripts de teste desempenham, podemos enumerar:

1. Permitir que os testes sejam executados de forma repetitiva e consistente, garantindo que as vulnerabilidades de segurança sejam identificadas e corrigidas com mais eficiência.
2. Facilitar a integração contínua e na entrega contínua, permitindo que os testes sejam executados automaticamente sempre que houver uma nova atualização no software. Isso é essencial para garantir que as mudanças feitas no código não introduzam novas vulnerabilidades e que o software continue seguro.
3. Automatizar tarefas de segurança, como a análise estática de código e a verificação de configuração de segurança, garantindo que o software seja desenvolvido de acordo com as práticas recomendadas de segurança. Isso ajuda a reduzir o tempo gasto em tarefas manuais e aumenta a eficiência geral do processo de segurança.

Em resumo, os scripts são importantes no contexto de DevSecOps porque permitem a automação de tarefas de segurança e testes, melhorando a eficiência e a eficácia do processo de desenvolvimento e segurança de software.

Estrutura de testes

São conjuntos de ferramentas, bibliotecas e convenções que fornecem uma estrutura padronizada para criar e executar testes automatizados de software. As estruturas de teste fornecem um ambiente consistente para testes, permitindo que desenvolvedores e testadores criem e executem testes de maneira mais simples e rápida.

As estruturas de teste são uma parte fundamental do processo de desenvolvimento de software, especialmente em um ambiente DevSecOps, em que a automação e a orquestração de testes são essenciais para garantir a segurança e qualidade do software. As estruturas de teste podem ser usadas para testar desde pequenos módulos de código até sistemas inteiros, garantindo que o software seja testado em todas as etapas do processo de desenvolvimento.

Existem várias estruturas de teste disponíveis para diferentes linguagens de programação e tecnologias. Conheça agora as estruturas de teste mais populares!

- JUnit para Java
- NUnit para .NET
- PyTest para Python
- Jasmine para JavaScript

O uso de estruturas de teste ajuda a garantir que o software seja testado de maneira consistente e confiável, o que pode aumentar a eficiência do processo de desenvolvimento e melhorar a qualidade do software final.

Existem diversas estruturas de testes que são importantes para a implementação de DevSecOps. Vejamos algumas das mais relevantes!

Testes de unidade
Verificam se cada componente do software está funcionando corretamente de forma isolada, ou seja, testam unidades individuais de código.

Testes de integração
Garantem que diferentes módulos ou componentes do software trabalhem juntos corretamente, integrando as diferentes partes do sistema.

Testes de aceitação
Validam se o software atende aos requisitos do usuário e se funciona conforme o esperado.

Testes de performance
Medem o desempenho do software em relação a tempos de resposta, uso de recursos, capacidade de carga, entre outros aspectos.

Testes de segurança
Verificam a existência de vulnerabilidades no software, como falhas de autenticação, injeção de código, cross-site scripting, entre outras.

Testes de conformidade
Avaliam se o software está em conformidade com leis, normas e regulamentos aplicáveis, como a Lei Geral de Proteção de Dados (LGPD), PCI-DSS, HIPAA, entre outros.

É importante destacar que a escolha das estruturas de testes depende das necessidades específicas de cada projeto e das características do software em questão.

Verificando o aprendizado

Questão 1

A automação é uma tecnologia amplamente utilizada no processo de desenvolvimento de software e em outras áreas. Sobre sua definição, podemos afirmar que automação é

A

o conceito que se refere à capacidade de coordenar diferentes ferramentas e processos de automação para criar uma cadeia de eventos contínua.

B

a tecnologia que permite tarefas repetitivas serem efetuadas de maneira automatizada; por exemplo, a realização de testes funcionais e de segurança, compilação de código, publicação de artefatos em ambientes etc.

C

a tecnologia que permite facilitar a colaboração entre os membros da equipe de desenvolvimento, garantindo que o código-fonte seja frequentemente integrado ao repositório principal e o trabalho ocorra a partir de uma base atualizada frequentemente.

D

uma tecnologia integrante do processo de desenvolvimento do software e permeia por todas as fases de seu ciclo de desenvolvimento, desde a ideação até sua operação e evolução.

E

a tecnologia que faz parte de DevOps e nada tem a ver com DevSecOps.

A alternativa B está correta.

Automação é uma tecnologia que permite que tarefas repetitivas e rotineiras sejam executadas de forma automatizada, aumentando a eficiência e a qualidade do processo de desenvolvimento de software. A opção B é a única que destaca especificamente a execução automatizada de tarefas, como testes funcionais e de segurança, compilação de código, publicação de artefatos em ambientes, verificação de vulnerabilidades e provisionamento de recursos, que são exemplos comuns de uso da automação em projetos de desenvolvimento de software. As outras opções abordam aspectos importantes da automação, mas não definem o seu conceito de forma precisa.

Questão 2

A orquestração é um conceito importante na cultura DevOps e DevSecOps. Assinale a alternativa que apresenta a definição correta para orquestração.

A

É o conceito que se refere à capacidade de coordenar diferentes ferramentas e processos de automação para criar uma cadeia de eventos contínua.

B

É a tecnologia que permite tarefas repetitivas serem efetuadas de maneira automatizada; por exemplo, a realização de testes funcionais e de segurança, compilação de código, publicação de artefatos em ambientes, verificação de vulnerabilidades, provisionamento de recursos etc.

C

É a tecnologia que permite facilitar a colaboração entre os membros da equipe de desenvolvimento, garantindo que o código-fonte seja frequentemente integrado ao repositório principal permitindo que todos os desenvolvedores trabalhem a partir de uma base atualizada frequentemente.

D

É uma tecnologia integrante do processo de desenvolvimento do software e que permeia por todas as fases de seu ciclo de desenvolvimento, desde a ideação até sua operação e evolução.

E

É a tecnologia que faz parte de DevOps e nada tem a ver com DevSecOps.

A alternativa A está correta.

A orquestração é uma técnica muito utilizada em arquiteturas distribuídas, em que é necessário gerenciar a execução de diversas tarefas, processos e serviços, de forma a garantir que as dependências entre eles sejam satisfeitas e que a execução ocorra de maneira sequencial ou paralela, conforme a necessidade.

Conceitos de modelagem de ameaças

Assista ao vídeo e fique por dentro da modelagem de ameaças em DevSecOps. Isso ajudará você a identificar e prever possíveis vulnerabilidades e ameaças de segurança com antecedência.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Modelagem de ameaças

É uma técnica utilizada em DevSecOps para identificação e avaliação de possíveis ameaças à segurança em um sistema ou aplicação. A técnica busca a criação de um modelo que represente todos os componentes e processos do sistema e permita avaliar as possíveis ameaças que podem surgir em cada etapa.

A modelagem de ameaças é essencial no DevSecOps, pois ajuda a identificar antecipadamente possíveis vulnerabilidades e ameaças de segurança. Isso ajuda a reduzir o risco de serem explorados por pessoas mal-intencionadas. Ao aplicar essa técnica, é possível descobrir e compreender melhor os pontos fracos do sistema, permitindo tomar medidas para corrigir ou reduzir essas vulnerabilidades.

> A modelagem de ameaças pode e deve ser realizada nas diversas etapas do ciclo de vida do desenvolvimento de software, desde a concepção até a entrega em produção da solução.

Além disso, essa técnica pode ser integrada com outras práticas de segurança, como testes de penetração e análise de estática de código-fonte, fornecendo assim uma visão ainda mais abrangente das possíveis ameaças à segurança.

Existem diversas ameaças que um sistema pode sofrer. Vamos entendê-las!

Spoofing

É o tipo de ataque em que uma pessoa ou um sistema assume a identidade de outra pessoa ou organização por meio da falsificação de dados para enganar uma vítima e assim ganhar sua confiança para ter acesso e realizar ações maliciosas. Um exemplo de spoofing é o ataque phishing, em que o atacante envia um e-mail fingindo ser uma empresa ou indivíduo, a fim de obter informações sensíveis de sua vítima.

Tampering

É a ameaça com o objetivo de alterar dados, códigos ou configurações de um sistema criando assim uma vulnerabilidade para que o atacante explore e atinja seu objetivo que pode ser o roubo de dados ou até mesmo prejudicar o funcionamento do sistema.

Repudiation

É uma ameaça à integridade dos dados dos sistemas. Ocorre normalmente quando um usuário ou sistema nega ter executado uma ação ou transação específica, e o sistema afetado não possui mecanismos que garantam a rastreabilidade das ações dos usuários para garantir a autenticidade das transações realizadas.

Exploitability

É a ameaça em que o atacante explora uma vulnerabilidade do sistema por meio da injeção de códigos que possam comprometer a integridade do sistema, permitindo assim acesso indevido, utilizando técnicas como SQL Injection, Buffer Overflow etc.

Para tentar prever, monitorar e impedir que ameaças sejam exploradas existem diversas metodologias e ferramentas disponíveis para a modelagem de ameaças, vejamos!

Dread

(Damage, reproducibility, exploitability, affected users and discoverability): Tem como objetivo a análise de risco qualitativa para priorizar as ameaças com base em cinco critérios que compõem a sua sigla, sendo muito utilizada para avaliar a gravidade das ameaças. As ameaças são então categorizadas em uma escala entre 5 e 15 e, com base na nota que o item tiver, ele pode ser considerado de alto, médio ou baixo risco.

Stride

(Spoofing, tampering, repudiation, information disclosure, denial of service e elevation of privilege): Tem como objetivo identificar ameaças em cada um dos seis tipos de ataques que compõem a sua sigla, sendo uma metodologia amplamente utilizada em grandes corporações provedoras de soluções em tecnologia, como a Microsoft.

Pasta

(Process for attack simulation and threat analysis): Tem como objetivo ter amplitude na análise das ameaças através de um contexto mais amplo que a Dread e a Strike, considerando fatores como os atores das ameaças, ativos, vulnerabilidades e impacto potencial caso uma vulnerabilidade seja explorada.

Trike

(Threats, risks, identities, knowledge and environment): É um processo de modelagem de ameaças de código aberto e tem como objetivo analisar ameaças baseada no processo de auditoria de segurança cibernética.

Cada metodologia tem sua própria abordagem e técnica, mas todas compartilham o objetivo comum de identificar e mitigar ameaças à segurança em sistemas e aplicações.

Controles de segurança e resposta a incidentes

Confira os principais controles de segurança que você deve implementar para reduzir riscos e proteger seus sistemas contra ameaças e vulnerabilidades, seguindo normas internacionais.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Controles de seguranças

São medidas que devem ser implementadas para mitigar riscos e proteger sistemas contra possíveis ameaças e vulnerabilidades. Os principais controles de segurança conhecidos são regidos por normas internacionais. São elas:

ISO/IEC 27001:2013

Trata do tema de sistema de gestão de segurança da informação.

SO/IEC 27002/2022

Trata do tema segurança da informação, cibersegurança e proteção de privacidade – Controles de segurança da informação.

ISO/IEC 27005/2018

Trata do tema gestão de risco de segurança da informação.

ISO/IEC 27017/2015

Trata do tema código de prática para segurança da informação em nuvem.

Outras normas também são amplamente difundidas e implementadas pelas empresas com o objetivo de desenvolver e implementar camadas adicionais de segurança aos sistemas, podendo destacar a norma **Nist SP 800-53**, que determina uma estrutura de controle de segurança desenvolvida pelo Instituto Nacional de Padrões e Tecnologia (NIST) dos Estados Unidos e a CIS Controls, que á uma lista com 20 controles de segurança desenvolvida pelo Center for Internet Security (CIS).

Observe alguns exemplos de controles de segurança!

Controle de acesso

Todo sistema que gerencia dados sensíveis deve ter mecanismos de autenticação, autorização e criptografia, limitando o acesso ao sistema e seus dados apenas a pessoas autorizadas. Isso é mandatório.

Monitoramento

Todo sistema deve ter implementado mecanismos de armazenamento de log para seu monitoramento, permitindo assim sua análise e detecção de eventos suspeitos de exploração de possíveis vulnerabilidades.

Backup e recuperação
Mecanismos de backup e recuperação devem ser implementados para garantir disponibilidade e integridade das informações críticas dos sistemas.

Controle de segurança de infraestrutura
Todo sistema deve utilizar firewalls e mecanismos de detecção de intrusões de rede, proteção contra malwares e ameaças externas.

Planejamento de resposta a incidentes

Parte fundamental da gestão de segurança da informação, o planejamento de resposta a incidentes apresenta o plano de ação para tratar incidentes de segurança, caso ocorram. Tem como objetivo minimizar os danos causados pelo incidente, restaurar o pleno funcionamento dos sistemas afetados o mais rápido possível, além de investigar a causa raiz que permitiu que o incidente ocorresse, identificando solução e melhorias nos processos afetados para garantir que o incidente não ocorra novamente no futuro.

As principais normas que regem o tema de resposta a incidentes incluem:

ISSO/IEC 279035

Information Technology – Security Techniques – Information Security Incident Management

NIST SP 800-61r2

Computer Security Incident Handling Guide

SANS Institute

Incident handling Step-by-Step Guide

Todas as normas determinam um conjunto de técnicas e planos para o a gestão efetiva de incidentes para as empresas. Mas todas têm em comum um conjunto de ações que devem ser contempladas em seus planos. Confira!

1
Notificação e comunicação
Procedimento para determinar quais pessoas e áreas devem ser notificadas e comunicadas na ocorrência de um incidente, e como essa comunicação deve ser realizada para as partes interessadas (funcionários, clientes, reguladores etc.).

2
Identificação e isolamento
Procedimento para que seja realizada a identificação dos sistemas afetados pelo incidente e o pleno isolamento destes para que outros sistemas não sejam afetados, com o objetivo de limitar o impacto do incidente.

3 Recuperação de desastres

Procedimento para restaurar as operações normais do negócio alinhado a procedimento coleta, análise e preservação de evidências do incidente para ações de prevenção futura.

Verificando o aprendizado

Questão 1

A modelagem de ameaças é importante para a implementação da segurança na cultura DevSecOps. Ela pode ser definida como

A

a capacidade de coordenar diferentes ferramentas e processos de automação para criar uma cadeia de eventos contínua.

B

medida que deve ser implementada para mitigar riscos e proteger sistemas contra possíveis ameaças e vulnerabilidades.

C

uma técnica que elabora o plano de ação para tratar incidentes de segurança, a fim de minimizar os danos causados, restaurar o pleno funcionamento dos sistemas afetados e investigar a causa raiz.

D

uma técnica para identificação e avaliação de possíveis ameaças à segurança em um sistema ou aplicação, que busca um modelo que represente todos os componentes e processos do sistema e permita avaliar as possíveis ameaças em cada etapa.

E

um documento estabelecido por governos e organizações a fim de definir requisitos e diretrizes de segurança para as empresas, com normas técnicas e práticas recomendadas, para proteger operações, sistemas e dados das empresas.

A alternativa D está correta.

A modelagem de ameaças é uma técnica de identificação e avaliação de possíveis ameaças à segurança em um sistema ou aplicação, que é utilizada em DevSecOps. Essa técnica consiste na criação de um modelo que representa todos os componentes e processos do sistema e permite avaliar as possíveis ameaças que podem surgir em cada etapa. A partir desse modelo, é possível identificar as vulnerabilidades e implementar medidas para mitigar os riscos e proteger o sistema contra possíveis ameaças.

Questão 2

Qual é a definição correta de controles de segurança?

A

São ferramentas e processos de automação que trabalham juntos para criar uma cadeia de eventos contínua.

B

São medidas que devem ser implementadas para reduzir riscos e proteger sistemas contra ameaças e vulnerabilidades.

C

Abarcam um plano de ação para tratar incidentes de segurança, com o objetivo de minimizar danos e restaurar o pleno funcionamento dos sistemas afetados o mais rápido possível.

D

São uma técnica utilizada em DevSecOps para identificar e avaliar possíveis ameaças à segurança em um sistema ou aplicação.

E

São documentos estabelecidos por governos e organizações com requisitos e diretrizes de segurança que as empresas devem seguir.

A alternativa B está correta.

Controles de segurança são medidas de segurança que devem ser implementadas em um sistema para minimizar os riscos e protegê-lo contra possíveis ameaças e vulnerabilidades. Eles são responsáveis por garantir que o sistema esteja em conformidade com as políticas de segurança estabelecidas e são implementados com base em avaliações de riscos e ameaças. Dentre os tipos de controles de segurança estão: controles físicos, controles técnicos e controles administrativos.

Práticas em segurança da informação

Assista ao vídeo e aprenda sobre conformidade e governança em segurança da informação. Descubra a importância de ter processos claros, bem definidos e uma boa comunicação dentro da organização.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Conceitos de conformidade e governança

Conheça agora os conceitos fundamentais de segurança da informação!

Conformidade

Refere-se ao cumprimento das leis, regulamentos e padrões estabelecidos com o objetivo de garantir a segurança dos dados e informações.

Governança

Trata do conjunto de processos, políticas, normas e diretrizes que devem ser utilizados para gerenciar e controlar a segurança da informação de uma organização.

Esses conceitos são complementares, e para sua adequada implementação são necessários:

- Estabelecimento de processos claros e bem definidos
- Definição de papéis e responsabilidades
- Plena comunicação dentro da organização

Conformidade e governança possuem normas que indicam quais são os mecanismos e como deve ser feita a implementação nas empresas, tendo como referência a norma ISO/IEC 27001 – Sistema de gestão de segurança da informação e a ISO/IEC 27002 – Segurança da informação, cibersegurança e proteção de privacidade – Controles de segurança da informação, COBIT (Control Objectives for Information and Related Technology), ISO 38500 – Governança de TI e ITIL (Information Technology Infrastructure Library). Adicionalmente temos a Lei Geral de Proteção de Dados brasileira, que deve ser observada e atendida em sua plenitude.

A análise e os métodos de conformidade aplicáveis em cada empresa devem levar em consideração o contexto de negócio que ela está inserida. Uma empresa que atua no mercado financeiro terá um conjunto de regulação a ser atendido, diferente de uma empresa que atua na indústria de óleo e gás, que terá outro conjunto de regulação. Algumas regras serão comuns a todos os contextos de negócio e certos mecanismos serão comuns a todos. Observe alguns exemplos de mecanismos de conformidade!

PCI DSS

Padrão de segurança de dados do setor de cartões de pagamento. É um padrão que se aplica a empresas que vão ter algum processo de negócio de pagamento por cartões de crédito e/ou débito. Ele define os requisitos de segurança que as empresas devem seguir ao processar, armazenar ou transmitir dados de cartão de pagamento.

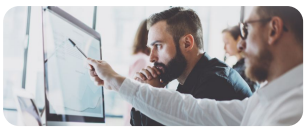
SOC 2

Service Organization Control 2 é um padrão de segurança que define as práticas recomendadas de segurança da informação para empresas que proveem serviços de computação em nuvem para outras empresas.

A governança apresenta também diversos mecanismos que devem ser considerados para sua implementação com o objetivo de garantir efetividade, eficiência e segurança das operações. Vejamos alguns dos principais mecanismos de governança!

Gerenciamento de risco

Identificação, avaliação e tratamento dos riscos que podem afetar a empresa, levando-se em consideração fatores, como: probabilidade, impacto, vulnerabilidade e ameaças.



Controles internos

Implementação de controles internos para garantir a integridade das operações, prevenção de fraudes e erros, proteção de ativos e cumprimento de regulamentações.

Compliance

Adoção de um programa de compliance que permita a empresa cumprir com as leis, regulamentações e políticas internas.

Responsabilidade corporativa

Promoção de uma cultura de responsabilidade corporativa, que envolve o comprometimento com a ética, integridade e transparência nos negócios.

Normas e diretrizes para segurança da informação

Assista ao vídeo e descubra a importância de regulamentos e padrões de segurança da informação para empresas, e veja exemplos de normas amplamente conhecidas pelo mercado, como ISO 27001, GDPR e PCI DSS.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Regulamentos e padrões de segurança

Durante todo o desenvolvimento do conteúdo, citamos diversos regulamentos e padrões que ajudam empresas a se prepararem para lidar com riscos e ameaças de segurança.

Regulamentos e padrões de segurança são documentos estabelecidos por governos e organizações com o objetivo de definir requisitos e diretrizes de segurança que as empresas devem seguir, fornecendo um conjunto de regras, normas técnicas e práticas recomendadas para garantir que as empresas implementem medidas de segurança adequadas para proteger operações, sistemas e dados.

É essencial lembrar que diferentes contextos de negócio em uma empresa estão sujeitos a regulamentos e padrões específicos. Por esse motivo, é crucial ter processos de negócio claramente definidos. A partir dessa definição, é possível construir ações para mitigar riscos de segurança da informação.

> A vantagem de se aplicar os padrões de segurança é que as empresas podem implementar um conjunto de práticas e técnicas que são amplamente difundidas e reconhecidas pela indústria como eficazes nos temas que abordam.

Além disso, implementando metodologias, mecanismos e práticas determinadas nos padrões de segurança garante-se que os processos de negócio e sistemas da empresa estão aderentes à prática de mercado, o que gera maior segurança e confiabilidade das empresas com seus clientes.

Confira **aqui** um breve resumo das principais normas e regulações que são amplamente conhecidas pelo mercado!

Aqui

https://stecine.azureedge.net/repositorio/00212ti/07621/docs/Resumo_de_normas_e_regulacoes.pdf

Auditoria de segurança: monitorando a proteção dos sistemas

Assista ao vídeo e descubra como a auditoria e o monitoramento podem ajudar a garantir a conformidade com as normas e procedimentos de segurança.

Conteúdo interativo

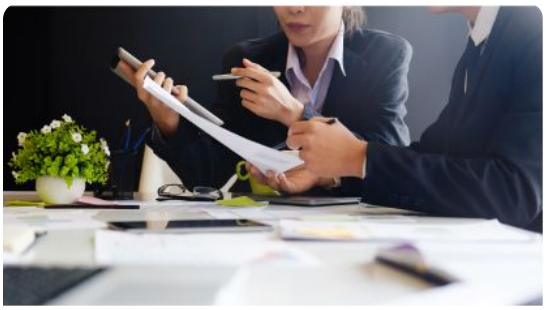
Acesse a versão digital para assistir ao vídeo.

Conceitos de auditoria e monitoramento

Acompanhe a definição desses conceitos!

Auditoria

É um processo de avaliação das atividades e processos de uma organização ou sistema para garantir a conformidade com normas e procedimentos estabelecidos, bem como identificar pontos de melhoria e riscos potenciais.



Monitoramento

É uma atividade contínua e sistemática que tem como objetivo avaliar o desempenho da empresa em relação a suas metas e objetivos, bem como verificar se políticas, processos e controles estão implementados e operando de maneira adequada.

Tanto auditoria quanto o monitoramento podem ser realizados de forma interna ou externa, sendo executadas por profissionais especializados capazes de fazer uma avaliação independente e imparcial da eficácia e eficiência dos controles e procedimentos internos da empresa. Em ambos os casos os resultados são documentos e relatórios que devem apontar os pontos de aderência e os pontos de falha de adoção de normas e procedimentos a serem utilizados para melhorar a conformidade, a governança e a gestão de riscos das empresas.

Existem diferentes modelos de auditoria e monitoramento que as empresas podem adotar, como:

- **Auditoria interna**: Realizada por uma equipe interna da empresa para avaliar o cumprimento de políticas, normas e procedimentos de segurança.
- **Auditoria externa**: Realizada por uma empresa terceirizada ou por um auditor independente para avaliar a eficácia dos controles de segurança da empresa.
- **Monitoramento de eventos de segurança**: Consiste na coleta e análise de informações de logs de sistemas, redes e aplicativos para identificar atividades suspeitas e possíveis violações de segurança.
- **Monitoramento de rede**: Consiste no monitoramento constante de tráfego de rede para identificar possíveis ameaças à segurança.
- **Testes de penetração**: Consiste na simulação de ataques cibernéticos para avaliar a segurança da infraestrutura de TI da empresa.
- **Avaliação de conformidade**: Consiste na avaliação regular da conformidade da empresa com padrões e normas de segurança, como PCI-DSS, ISO 27001, entre outros.
- **Auditoria de privacidade**: Consiste na avaliação da conformidade da empresa com leis e regulamentos de privacidade, como GDPR e LGPD.

Verificando o aprendizado

Questão 1

Qual a importância dos conceitos de conformidade e governança para a implementação de uma cultura de segurança em empresas que utilizam a abordagem DevSecOps?

A

Conformidade e governança são documentos estabelecidos por governos e organizações para definir requisitos e diretrizes de segurança que as empresas devem seguir, incluindo práticas de segurança adequadas para proteger operações, sistemas e dados.

B

Conformidade refere-se ao cumprimento das leis, regulamentos e padrões estabelecidos para garantir a segurança das informações, e governança trata do conjunto de processos, políticas, normas e diretrizes para gerenciar e controlar a segurança da informação de uma organização.

C

Conformidade e governança são processos de avaliação das atividades e processos de uma organização ou sistema para garantir conformidade com normas e procedimentos estabelecidos, bem como identificar pontos de melhoria e riscos potenciais.

D

Conformidade e governança são atividades contínuas e sistemáticas que têm como objetivo avaliar o desempenho da empresa em relação a suas metas e objetivos, bem como verificar se políticas, processos e controles estão implementados e operando adequadamente.

E

Conformidade e governança são disciplinas que contemplam a operação dos processos das empresas e nada têm a ver com DevSecOps.

A alternativa B está correta.

Conformidade e governança são conceitos fundamentais em segurança da informação e são de extrema importância para o sucesso de uma iniciativa DevSecOps. A conformidade refere-se ao cumprimento de leis, regulamentos e padrões estabelecidos com o objetivo de garantir a segurança dos dados e informações. A governança trata do conjunto de processos, políticas, normas e diretrizes que devem ser utilizados para gerenciar e controlar a segurança da informação de uma empresa. Ambos são essenciais para garantir a segurança dos sistemas e informações em uma iniciativa DevSecOps, garantindo que as práticas de segurança sejam aderentes aos requisitos regulatórios e padrões estabelecidos, e que os processos de segurança sejam eficazes e eficientes.

Questão 2

Sobre os regulamentos e padrões de segurança para o processo de DevSecOps, é correto afirmar que

A

são documentos estabelecidos por governos e organizações com o objetivo de definir requisitos e diretrizes de segurança que as empresas devem seguir, fornecendo um conjunto de regras, normas técnicas e práticas recomendadas para garantir que as empresas implementem medidas de segurança adequadas para proteger operações, sistemas e dados.

B

são conceitos fundamentais, em que a conformidade se refere ao cumprimento de leis e padrões estabelecidos para garantir a segurança de dados e informações, e a governança trata de processos, políticas, normas para gerenciar a segurança da informação de uma organização.

C

são processos de avaliação de atividades e processos de uma organização ou um sistema para garantir a conformidade com as normas e os procedimentos estabelecidos, bem como identificar pontos de melhoria e riscos potenciais.

D

são atividades contínuas e sistemáticas que têm como objetivo avaliar o desempenho da empresa em relação a suas metas e objetivos, bem como verificar se as políticas, os processos e controles estão implementados e operando de maneira adequada.

E

são disciplinas que contemplam a operação dos processos das empresas e nada têm a ver com DevSecOps.

A alternativa A está correta.

Os regulamentos e padrões de segurança são importantes para DevSecOps, pois fornecem um conjunto de diretrizes e requisitos que ajudam as empresas a implementarem medidas de segurança adequadas para proteger operações, sistemas e dados. Esses documentos estabelecem um conjunto de regras, normas técnicas e práticas recomendadas que ajudam as empresas a implementarem controles de segurança eficazes e a cumprir com as exigências regulatórias e de conformidade. Ao seguir esses regulamentos e padrões, as empresas podem aumentar a confiança dos clientes, evitar penalidades legais e melhorar a segurança geral do sistema.

Considerações finais

Chegando aqui, certamente, você teve uma boa base para determinar o que é DevSecOps e entender o porquê de essa prática de engenharia de software ser tão importante para acelerar a entrega de um software cada vez mais seguro.

Além disso, compreendeu que riscos e ameaças não são apenas do ponto de vista técnico, mas também do ponto de vista pessoal. Entendeu que existem diversas normas e padrões a serem adotados para garantir que os riscos de ameaças e vulnerabilidades sejam mitigados, diminuindo assim o dano potencial desses riscos.

Neste conteúdo, foi possível conferir a importância de se ter os processos de negócio que sua empresa atua bem mapeados, a fim de determinar de maneira clara quais regulações, normas e padrões devem ser implementados, juntamente com processos e governança claros para garantir que essas normas sejam devidamente atendidas.

Lembre-se ainda que auditoria e monitoramento são benéficos ao negócio, pois evidenciam de maneira proativa possíveis pontos de vulnerabilidade e melhoria.

Agora, você é mais do que capaz de identificar como o DevSecOps pode potencializar a segurança dos seus produtos de software e entender como ter uma cultura com foco em segurança pode ajudar a alavancar ainda mais o potencial do seu negócio. Esperamos que tenha tido um bom proveito na aplicação desse conhecimento!

Explore +

Confira as indicações de leitura que separamos especialmente para você!

- **DevSecOps: A Comprehensive Guide**, artigo disponível no portal Owasp. Vale conferir!
- **DevSecOps: Shifting Security Left with Automation**, artigo disponível no portal Red Hat. Imperdível!

Referências

ADKINS, H. *et al*. **Building Secure and Reliable Systems**: Best Practices for Designing, Implementing, and Maintaining Systems. 1. ed. Sebastopol: O'Reilly Media, 2020.

BRASIL. **Lei nº 13.709, de 14 de agosto de 2018**. Lei Geral de Proteção de Dados Pessoais (LGPD). Brasília, 2018.

CRISPIN, L.; GREGORY, J. **Agile Testing**: A Practical Guide for Testers and Agile Teams. 1. ed. Boston: Addison-Wesley Professional, 2009.

HUMBLE, J.; FARLEY, D. **Continuous Delivery**: Reliable Software Releases through Build, Test, and Deployment Automation. 1. ed. Boston: Addison-Wesley Professional, 2011.

ISO. **ISO/IEC 27001**: Information security management. Consultado na internet em: 25 maio 2023.

ISO. **ISO/IEC 27002**: Information Technology - Security Techniques - Code of Practice for Information Security Controls. Consultado na internet em: 25 maio 2023.

KIM, G. *et al*. **The DevOps Handbook**: How to Create World-Class Agility, Reliability, and Security in Technology Organizations. 1. ed. Portland: IT Revolution Press, 2016.

MICROSOFT. **Microsoft Security Development Lifecycle (SDL)**. Consultado na internet em: 25 maio 2023.

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. NIST. **Cybersecurity Framework**. Gaithersburg: NIST, 2018.

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. NIST. **Special Publication 800-53**: Security and Privacy Controls for Information Systems and Organizations. Gaithersburg: NIST, 2020.

OPEN WEB APPLICATION SECURITY PROJECT. OWASP. **Dread**. Consultado na internet em: 25 maio 2023.

PAYMENT CARD INDUSTRY DATA SECURITY STANDARD. PCI DSS. **PCI Security Standards Council**. Consultado na internet em: 25 maio 2023.

SHOSTACK, A. **Threat Modeling**: Designing for Security. 1. ed. Indianapolis: Wiley Publishing, 2014.