





OWASP
Open Web Application
Security Project

OWASP Chapter São Paulo

OWASP TOP 10 para QA'S

Facilitadora: Alessandra Martins



O JEITO FEMININO DE FALAR DE TI

Agenda:

Conceitos, Princípios e Técnicas

Pessoas, Papéis e Responsabilidades

Processos e Ferramentas

Cultura DevOps, Auxilio no Fluxo de Dados e Segurança

Mito vs Realidade

Compliance e Documentação de Testes

Como encaixar OWASP TOP 10 , Governança de Dados e Compliance

Referências

O teste é um processo de comparação entre o estado de um sistema ou aplicativo em relação a um conjunto de critérios.

Conceitos

Engenharia de Software

- “É uma disciplina de engenharia, cuja a meta é o desenvolvimento de sistemas de software com uma boa relação custo x benefício” SOMMERVILLE
- O principal objetivo da engenharia do software é auxiliar na construção de produtos de qualidade

Garantia da Qualidade

- O Software Quality Assurance ou Garantia de Qualidade de Software, refere-se a um conjunto de atividades que visa assegurar que todos os esforços serão feitos para garantir que os produtos de software tenham a qualidade desejada.
- O principal objetivo é garantir que o projeto emprega todos os processos e padrões necessários para atender aos requisitos.

Controle de Qualidade

- “Controle de qualidade é definido como os processos e métodos usados para monitorar o trabalho e os requerimentos envolvidos. É focado nas revisões e remoção de defeitos antes da entrega do produto. Controle de qualidade dever ser responsabilidade da unidade de produção do produto dentro da organização...” (p.64, COSTA, NETO, COSTA NETO, & JUNIOR, 2013)
- O principal objetivo do controle de qualidade é descobrir defeitos em produtos de trabalho gerados ao longo do projeto.

Conceitos



Qualidade de Software

- A qualidade é um processo sistemático que focaliza todas as etapas e artefatos produzidos com o objetivo de garantir a conformidade de processos e produtos, prevenindo e eliminando defeitos.
- O objetivo da qualidade de software é garantir a qualidade de um produto de software através da definição e normatização dos processos de desenvolvimento, garantindo um produto final que satisfaça as expectativas do cliente dentro do que foi acordado entre as partes.



Teste de Software

- “O teste de programas pode ser usado para mostrar a presença de defeitos, mas nunca para mostrar a sua ausência.” (Dijkstra)
- O principal objetivo do Teste de Software medir a qualidade do software em termos de defeitos encontrados, por características e requisitos funcionais ou não funcionais do software (funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade – ISO 9216).



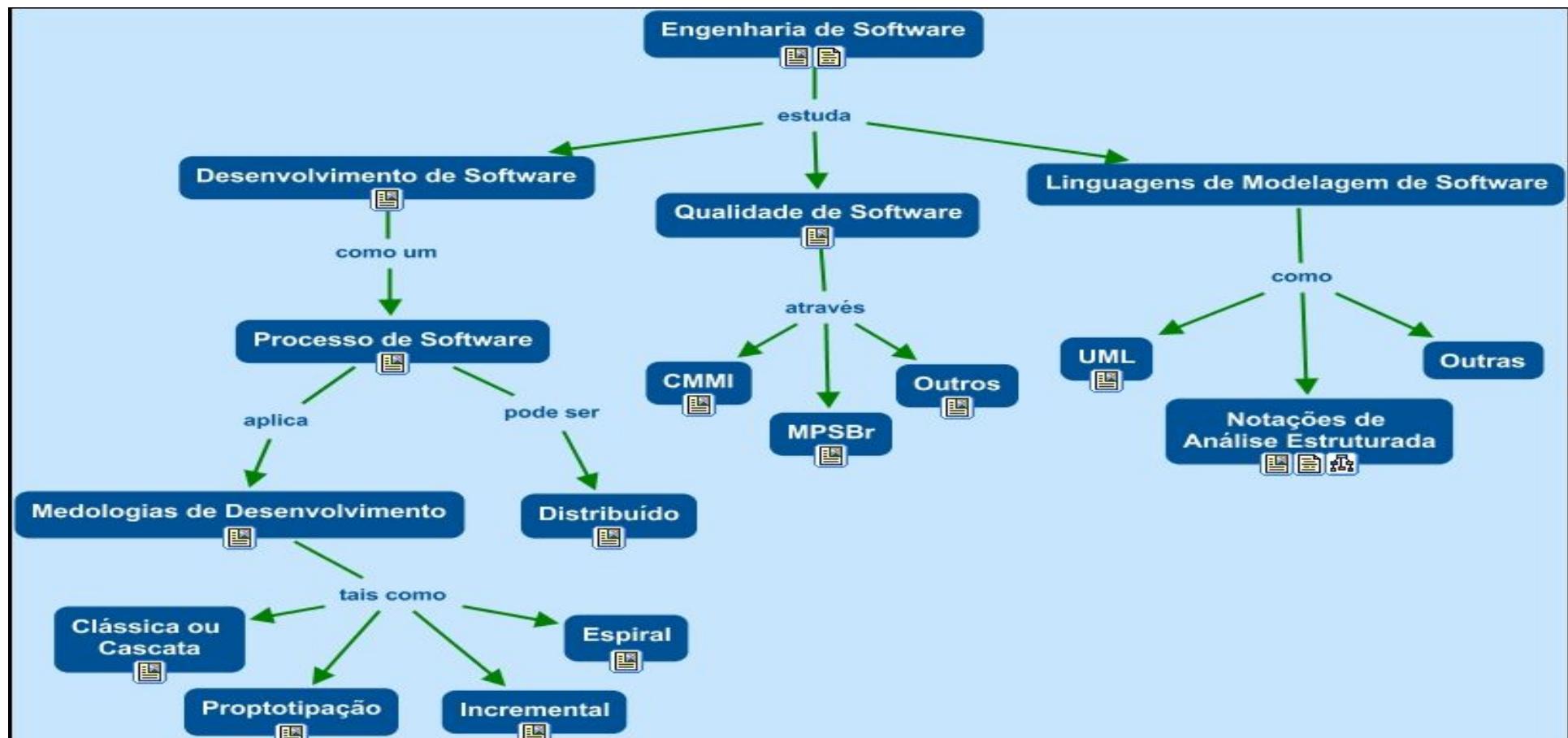
Segurança da Informação

- Segurança da Informação (SI) (Brostoff, 2004; Morris e Thompson, 1979; Sieberg, 2005; Smith, 2002) pode ser sumarizada como a proteção contra o uso ou acesso não-autorizado à informação, bem como a proteção contra a negação do serviço a usuários autorizados, enquanto a integridade e a confidencialidade dessa informação são preservadas. A SI não está confinada a sistemas de computação, nem à informação em formato eletrônico. Ela se aplica a todos os aspectos de proteção da informação ou dados, em qualquer forma.
- O principal objetivo da segurança é garantir tríade DIC (disponibilidade, Integridade e Confidencialidade), e ainda a autenticidade e não repúdio da informação.



Compliance

- No âmbito institucional e corporativo, Compliance é o conjunto de disciplinas para fazer cumprir as normas legais e regulamentares, as políticas e as diretrizes estabelecidas para o negócio e para as atividades da instituição ou empresa, bem como evitar, detectar e tratar qualquer desvio ou inconformidade que possa ocorrer.
- O termo compliance tem origem no verbo em inglês to comply, que significa agir de acordo com uma regra, uma instrução interna, um comando ou um pedido.





Conceitos

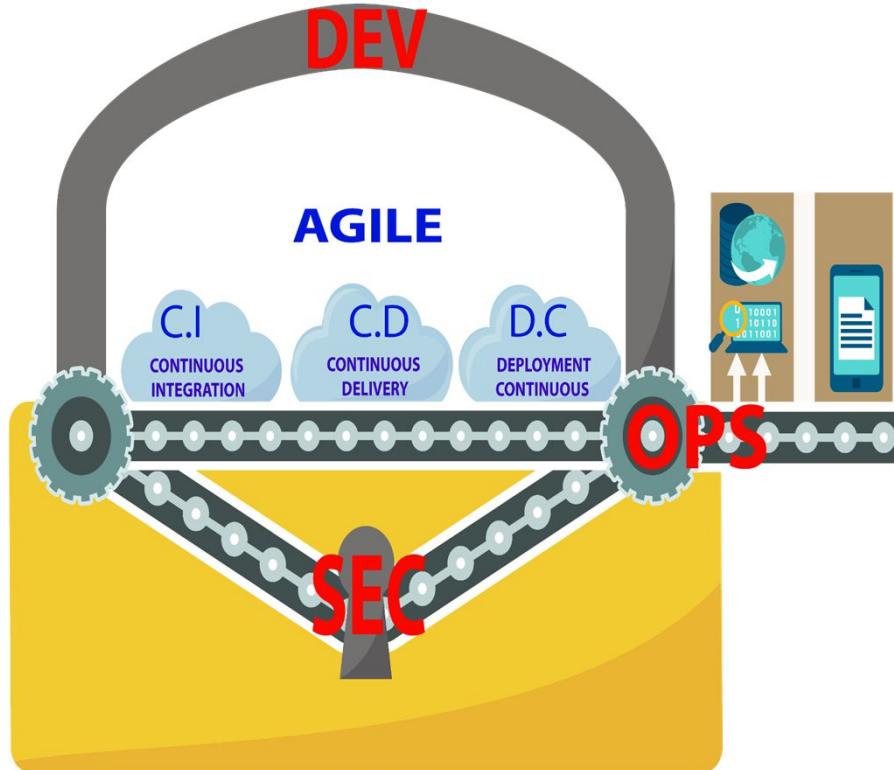
O que é DevOps ?

É um movimento cultural que muda o modo como os indivíduos pensam sobre o seu trabalho, valoriza a diversidade do trabalho realizado, suporta processos intencionais que aceleram a taxa pela qual as empresas realizam valor e mede o efeito da mudança social e técnica.

É uma maneira de pensar e um modo de trabalhar que permite que indivíduos e organizações desenvolvam e mantenham práticas de trabalho sustentáveis.

É um quadro cultural para compartilhar histórias e desenvolver empatia, permitindo que pessoas e equipes pratiquem seus ofícios de forma eficaz e duradoura.

Conceitos

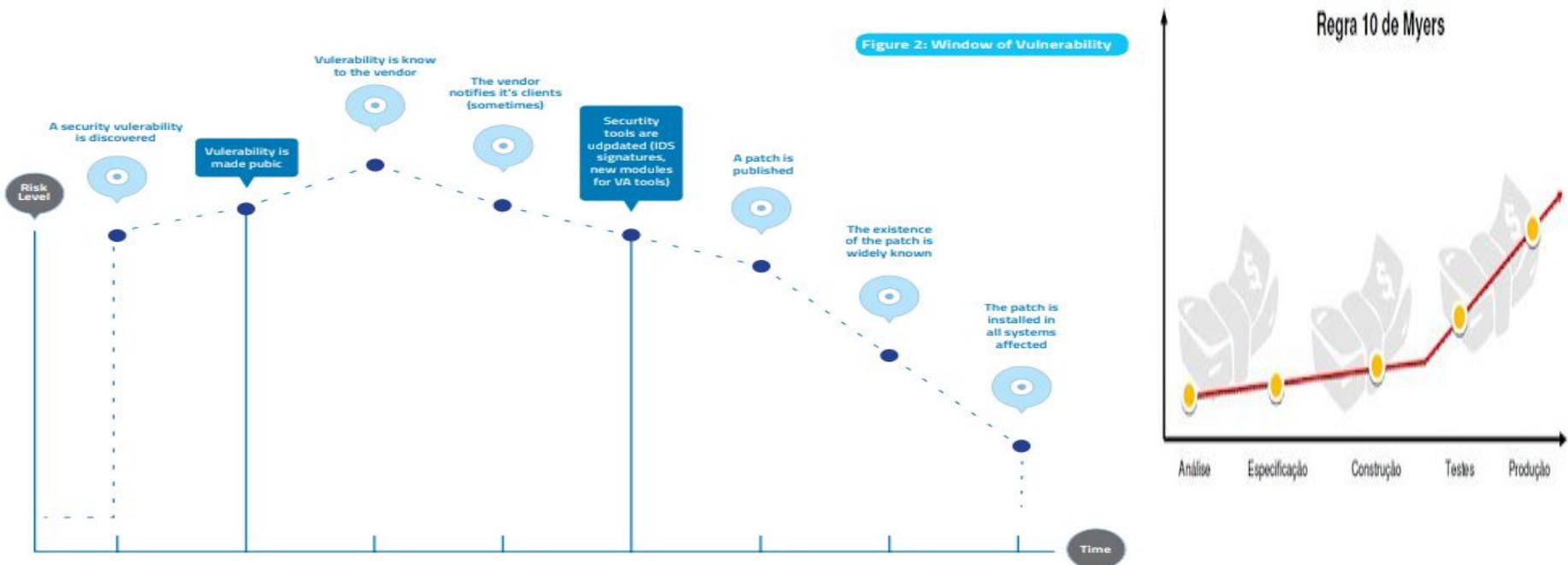


O que é DevSecOps ?

DevSecOps é um termo criado para descrever um conjunto de práticas para integração entre os times de Desenvolvimento de Software, Segurança e Operações e a adoção de processos automatizados para produção rápida e segura de aplicações e serviços

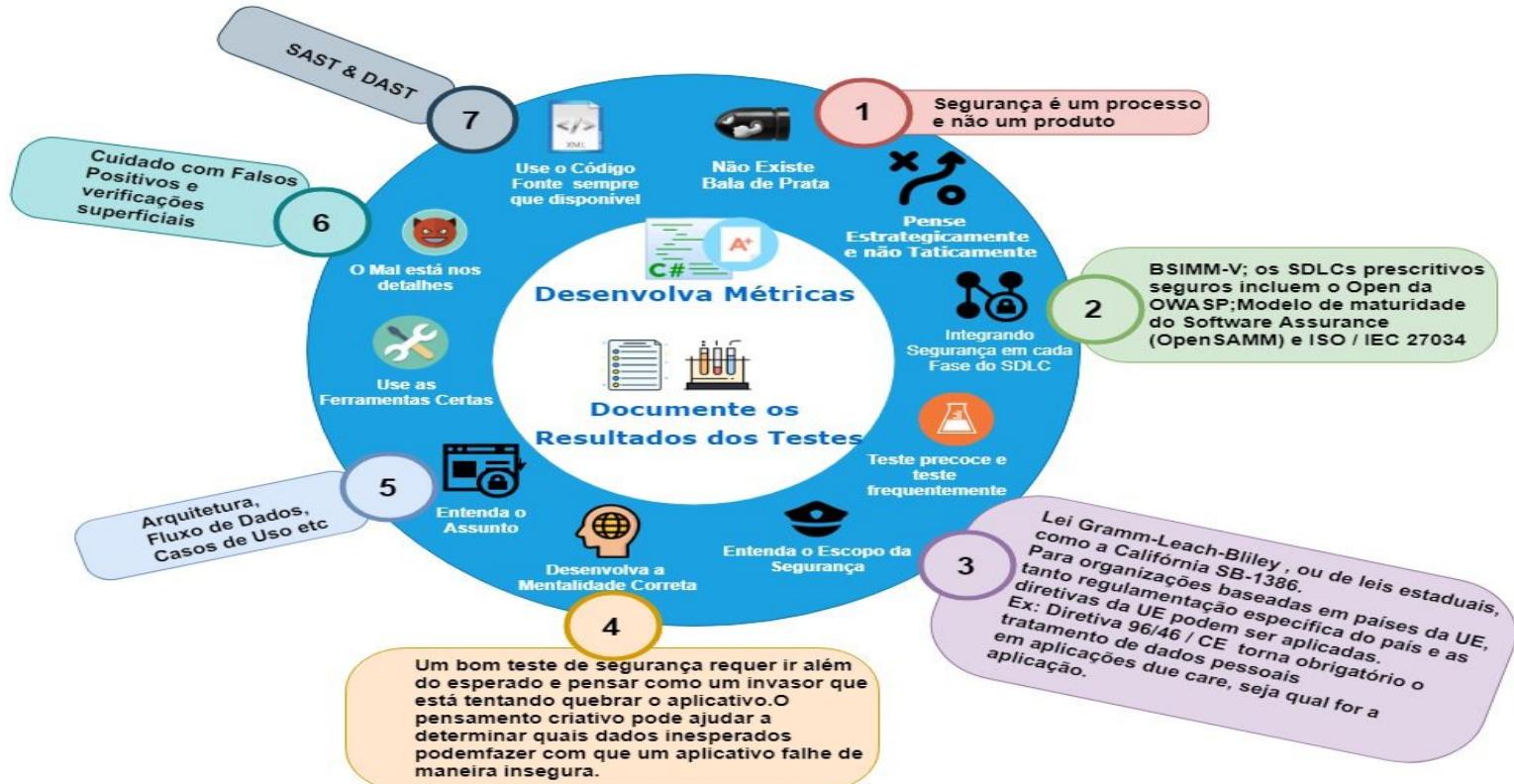
Princípios OWASP de testes

Custo da Falha & Janela de Vulnerabilidade:



Os 12 Princípios:

Princípios OWASP de testes



Alinhando Conceitos e Princípios

Uma Visão: qualidade de software alinhada a segurança

Pilares da Segurança da Informação

Integridade - ausência de alterações não autorizadas a um sistema, a uma ou mais informações;

Disponibilidade - a informação é acessível por usuários autorizados sempre que a solicitarem, está disponível;

Confidencialidade – diz respeito à ausência de divulgação não autorizada de informação, somente usuários autorizados podem visualizar uma informação;

Autenticidade - que diz respeito à origem do sistema ou informação fornecido, se são ou não genuínos.

Princípios de Teste e Qualidade de Software

1. Teste Demonstra a Presença de Defeitos
2. Teste Exaustivo é impossível
3. Testes dependem do contexto
4. A ilusão da ausência de erros
5. Teste Antecipado
6. Agrupamento de defeitos
7. Paradoxo do Pesticida

Processo: Conceitos OWASP Top 10 sobre testes

Um programa de teste efetivo deve ter componentes que testam:

Pessoas - para garantir que haja educação e conscientização adequadas



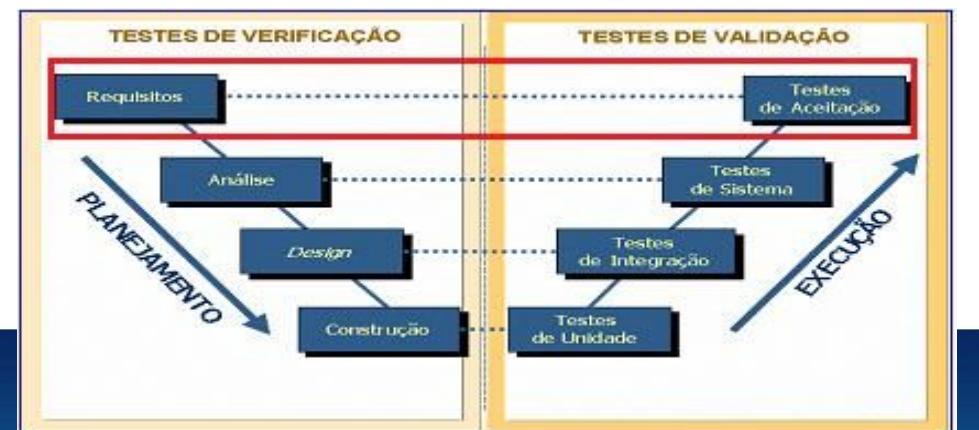
Processo - para assegurar que haja políticas e padrões adequados e que as pessoas saibam como seguir essas políticas



Tecnologia - para garantir que o processo tenha sido eficaz em sua implementação.



A menos que uma abordagem holística seja adotada, testar apenas a implementação técnica de um aplicativo não revelará vulnerabilidades de gerenciamento ou operacionais que possam estar presentes. Ao testar as pessoas, políticas e processos, uma organização pode detectar problemas que mais tarde manifestam-se em defeitos na tecnologia, erradicando assim erros cedo e identificar as causas de defeitos.



Técnicas Revisões e Inspeções Manuais

Inspeções manuais são revisões humanas que normalmente testam as implicações de segurança de pessoas, políticas e processos. As inspeções manuais também podem incluir inspeção de decisões tecnológicas, tais como como projetos arquitetônicos.

Vantagens:

- Não requer tecnologia de suporte
- Pode ser aplicado a uma variedade de situações
- Flexível
- Promove o trabalho em equipe
- No início do SDLC

Desvantagens:

- Pode ser demorado
- Material de suporte nem sempre disponível
- Requer pensamento e habilidade humanas significativas para serem eficazes

Técnicas Modelagem de Ameaças

Um modelo de ameaça, recomenda-se uma abordagem simples que siga o padrão NIST 800-30 [11] para avaliação de risco. Essa abordagem envolve:

- Decompondo o aplicativo - use um processo manual de inspeção para entender como o aplicativo funciona, seus ativos, funcionalidade e conectividade.
- Definir e classificar os ativos - classificar os ativos em ativos tangíveis e intangíveis e classificá-los de acordo com importância comercial.
- Explorar possíveis vulnerabilidades - sejam técnicas, operacional ou de gerenciamento.
- Explorando ameaças potenciais - desenvolva uma visão realista do potencial vetores de ataque da perspectiva de um invasor, usando ameaça cenários ou atacar árvores.
- Criar estratégias de mitigação - desenvolver controles de mitigação para cada uma das ameaças consideradas realistas.

A saída de um modelo de ameaça em si pode variar, mas normalmente é coleção de listas e diagramas.

Processo de Modelagem de Ameaças	
1	Identificar Ativos valiosos que devem ser protegidos
2	Criar uma visão geral da Arquitetura
3	Decompor a Arquitetura da Aplicação para criar camadas e perfis segurança
4	Identificar Ameaças
5	Documentar Ameaças
6	Classificar Ameaças

Vantagens:

- Visão prática do atacante do sistema
- flexível
- No início do SDLC

Desvantagens:

- Técnica relativamente nova
- Modelos de boa ameaça não significam automaticamente um bom software

Técnicas

Revisão de Código Fonte

A revisão do código-fonte é o processo de verificar manualmente a origem código de um aplicativo da web para problemas de segurança. Muitas vulnerabilidades sérias de segurança não podem ser detectadas com qualquer outra forma de análise ou teste.

Exemplos de problemas que são particularmente propícios para serem encontrados através de revisões de código fonte incluem problemas de simultaneidade, falhas na lógica de negócios, problemas de controle de acesso e fraquezas criptográficas, bem como backdoors, cavalos de Tróia, easter eggs, bombas-relógio, bombas lógicas e outras formas de código malicioso. Esses problemas geralmente se manifestam como as vulnerabilidades mais prejudiciais sites da web.

Vantagens:

- Completude e eficácia
- Precisão
- Rápido (para revisores competentes)

Desvantagens:

- Requer desenvolvedores de segurança altamente qualificados
- Pode perder problemas em bibliotecas compiladas
- Não é possível detectar erros em tempo de execução facilmente
- O código-fonte atualmente implementado pode diferir do código sendo analisado

Técnicas Pentest – Penetration Teste

O teste de penetração tem sido uma técnica comum usada para testar segurança de rede por muitos anos. Também é comumente conhecido como testes de caixa preta ou hacking ético. O teste de penetração é essencialmente a “arte” de testar um aplicativo em execução remotamente para encontrar vulnerabilidades de segurança, sem conhecer o funcionamento interno do aplicativo em si. Normalmente, a equipe de teste de penetração tem acesso a um aplicativo como se fossem usuários. O testador age como um invasor e tenta encontrar e explorar vulnerabilidades.

Vantagens:

- Pode ser rápido (e, portanto, barato)
- Requer um conjunto de habilidades relativamente menor que a revisão do código-fonte
- Testa o código que está sendo realmente exposto

Desvantagens:

- Tarde demais no SDLC
- Teste de impacto frontal apenas.

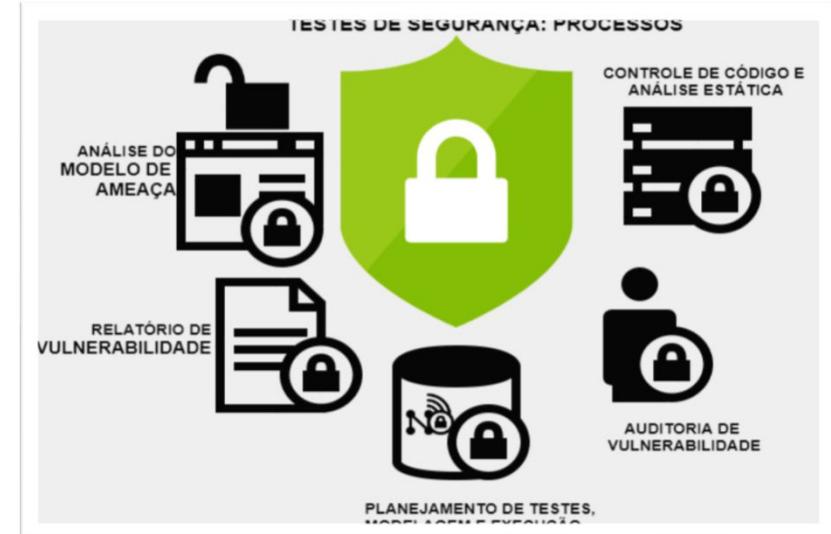
Técnicas

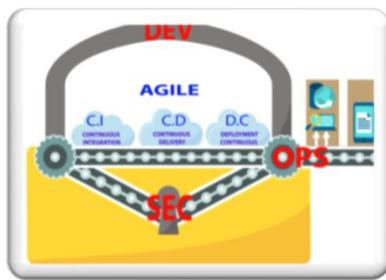
Práticas Envolvidas

A validação de segurança pode ser realizada em duas fases:

Estática –(SAST) tenta localizar falhas inseridas durante o desenvolvimento do projeto, como um estado não-alcançável ou possíveis erros humanos introduzidos no código. Nesse caso são utilizados métodos de análise estática (ex.: inspeção de código, analisadores de vulnerabilidade estáticos), ou prova de teorema, os quais não necessitam executar o sistema.

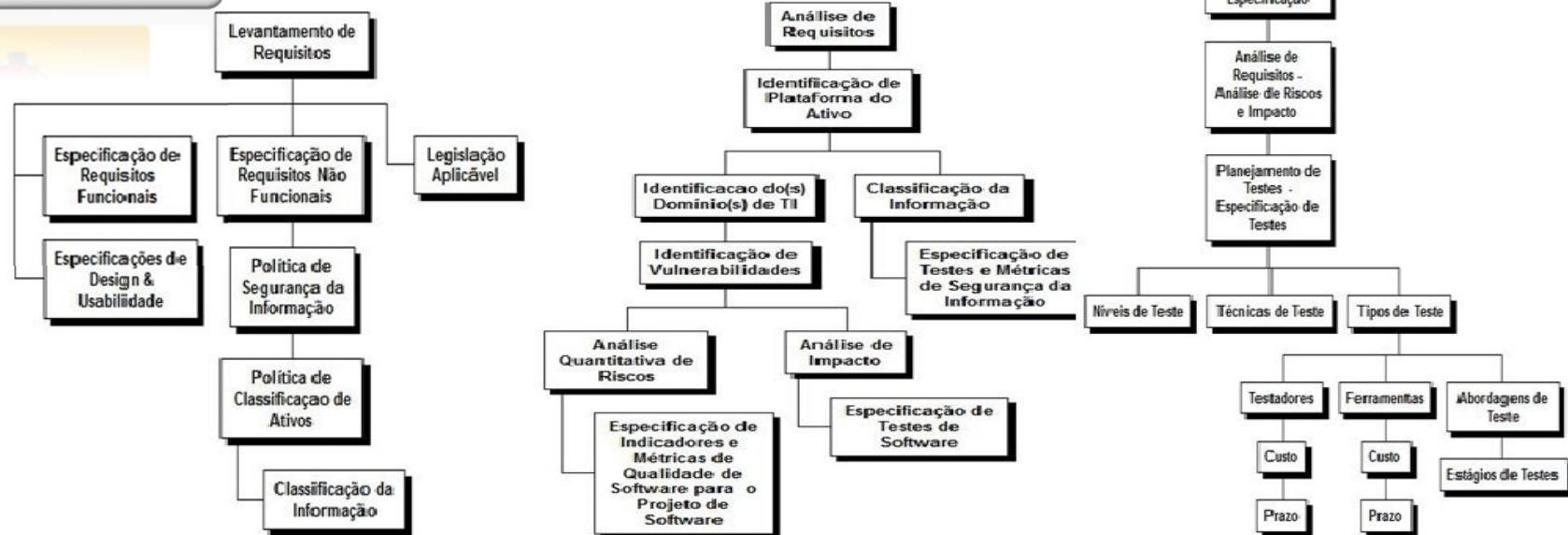
Dinâmica – (DAST) se foca na verificação da implementação durante sua execução, verificar o sistema exercitando seu código, onde entradas reais são fornecidas para verificar os mecanismos de segurança.





Técnicas: Modelagem de Testes

Etapas Modelo V- para Modelagem de Plano e Abordagem de Testes

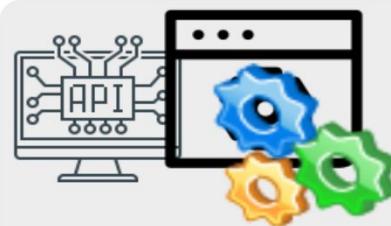


Técnicas

Contexto: Inserindo Testes de Segurança no Ciclo de Vida de Desenvolvimento de uma Aplicação

A parte mais visível do teste é a execução

O Teste de Segurança tem como meta garantir que o funcionamento da aplicação se comporta adequadamente mediante as mais diversas tentativas ilegais de acesso, visando possíveis vulnerabilidades. Para isso, testa-se todos os mecanismos de proteção embutidos na aplicação de fato a protegerão de acessos indevidos.



Testes de Software
Software Testing



Teste de Vulnerabilidades



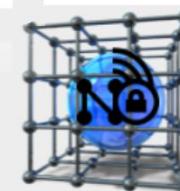
Pen Teste



Mobile



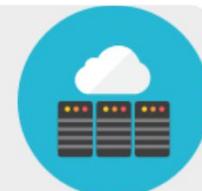
Web



Segurança de Redes



Code Review



Cloud App

Técnicas:

Modelagem

**Dimensões de Teste: Confiança, Funcionalidade e
Performance;**

Processos de Teste:

Verificação: Nós Construímos Corretamente o Sistema?

Validação: Nós Construímos o Sistema Correto?

Processo de Segurança:

Nós Construímos o sistema pensando na Segurança?

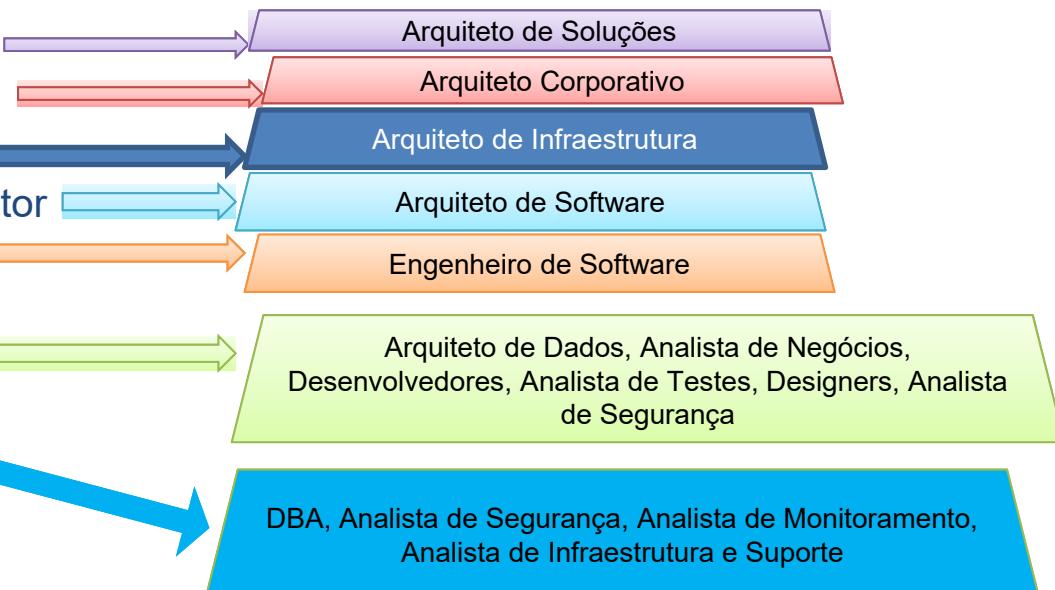
Nós validamos a segurança construída para o sistema?

Pessoas, Papéis e Responsabilidades

Sugestão de Composição de Papéis para Implantação DevOps+SEC :

Papéis:

1. Process Master (Scrum Master)
2. Service Master (Product Owner)
3. DevOps Engineer
4. Gatekeeper – Release Coordinator
5. Reliability Engineer (opcional)
6. Time Desenvolvimento (Dev, QA,DBA)
7. Time de Operação

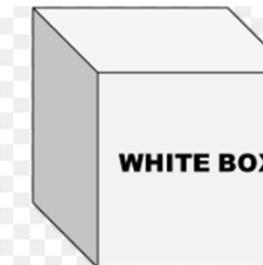
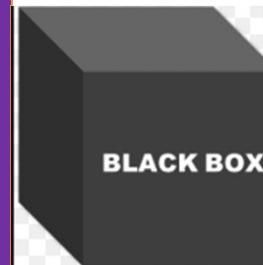


Pessoas, Papéis e Responsabilidades

Fuzzy -> Técnica para Injeção de falhas Escalável funciona com qualquer linguagem Versátil Host ou Rede

Especialista/ Analista de Segurança da Informação

- Capacidade Técnica - uso de Ferramentas;
- Conhecimentos de Segurança para Testar o Software;
- Análise de Vulnerabilidades;
- Modelagem de Cenários e Ameaças para Testes;
- Conhecimentos de Auditoria,
- Monitoria



- Pouca ou nenhuma informação fornecida;
- Vulnerabilidades mais críticas
- Determina o possível impacto na operação.
- Algumas informações são compartilhadas, mas de forma restrita;
- Combina as metodologias do Black e White box.
- Toda informação sobre o escopo é fornecida;
- Identificar todo tipo de vulnerabilidades;
- Análise de eficácia das ferramentas de controle.

Pessoas, Papéis e Responsabilidades

O papel do QA e algumas coisas mais:

QA não é Analista de Segurança, mas pode ser Capacitado e Colaborar com a Segurança

- Arquitetura de Teste –
- Framework e Ferramentas
- Automação de Testes com Scripts
- Escrita e Execução de Casos e Cenários de Testes Automatizados e Manuais

Cenários de Teste – US-BDD e Workflow

Teste Manual

Teste Exploratório

Perfil/ Papel do QA:



Análise de Protótipos

Revisão de Requisitos

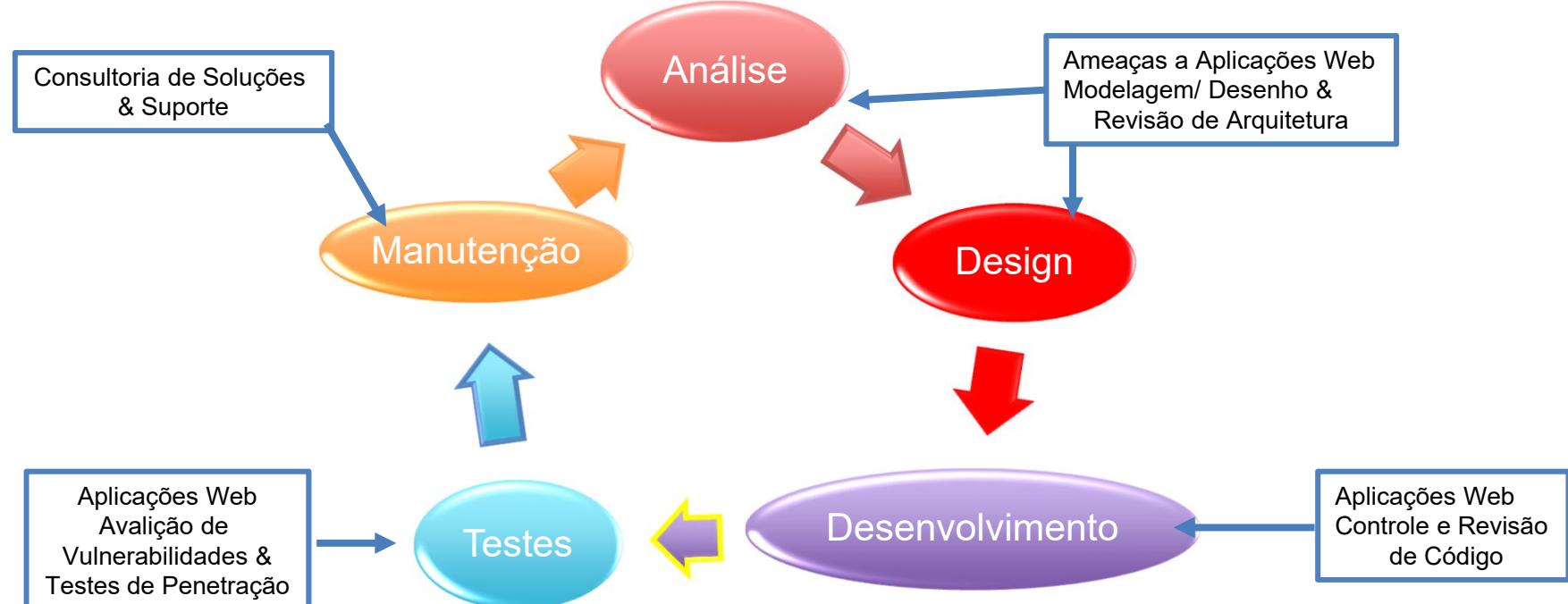
Casos de Teste

OWASP.ORG

Processo: Automação de Testes



Processo: Testes de Segurança



Processo e Ferramentas

Análise ou Pentest?

Análise de Vulnerabilidades

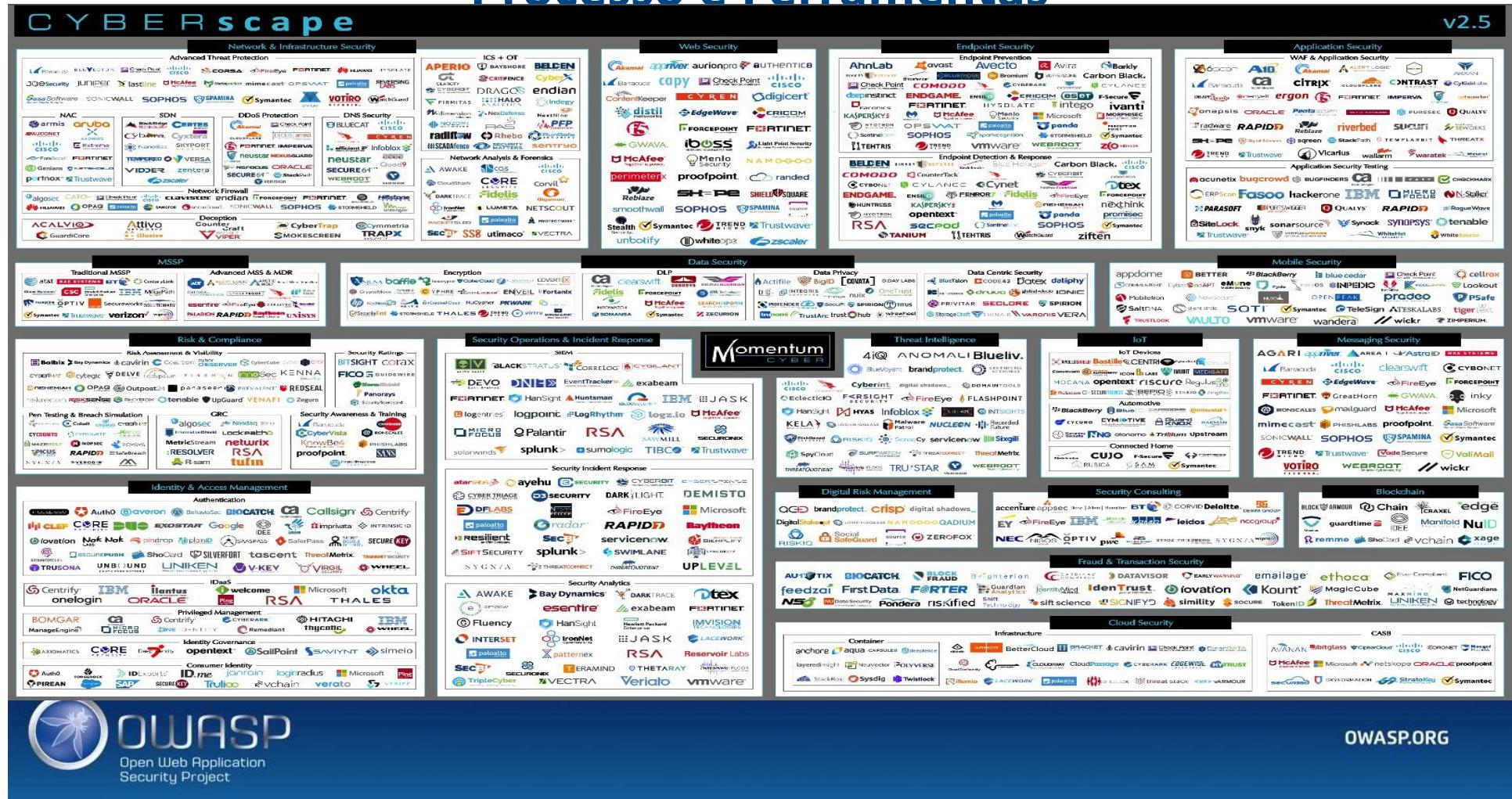
Indica possíveis vulnerabilidades em sua rede, sem necessariamente explorá-las. Muitas avaliações usam uma ferramenta de scan para identificar possíveis brechas em sistemas e políticas de segurança. Então, a ferramenta classifica por nível de impacto as vulnerabilidades encontradas em seu ambiente.



Pentest

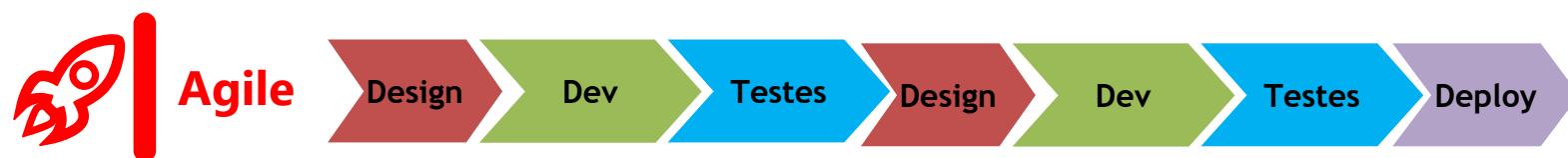
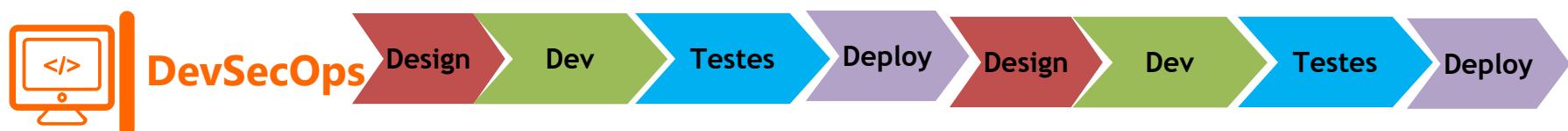
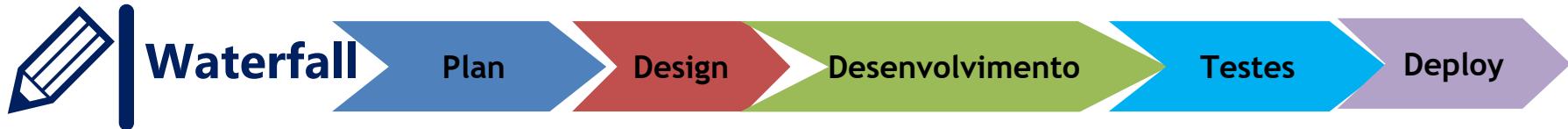
Consiste em uma avaliação mais extensiva, mais recomendada para organizações que já possuem uma postura de segurança madura. O objetivo do teste de penetração é identificar exploits dentro da rede ou aplicativos que tentam obter acesso a dados sensíveis. Com o pentest, também é possível mostrar o impacto financeiro de possíveis exploits em seu ambiente.

Processo e Ferramentas

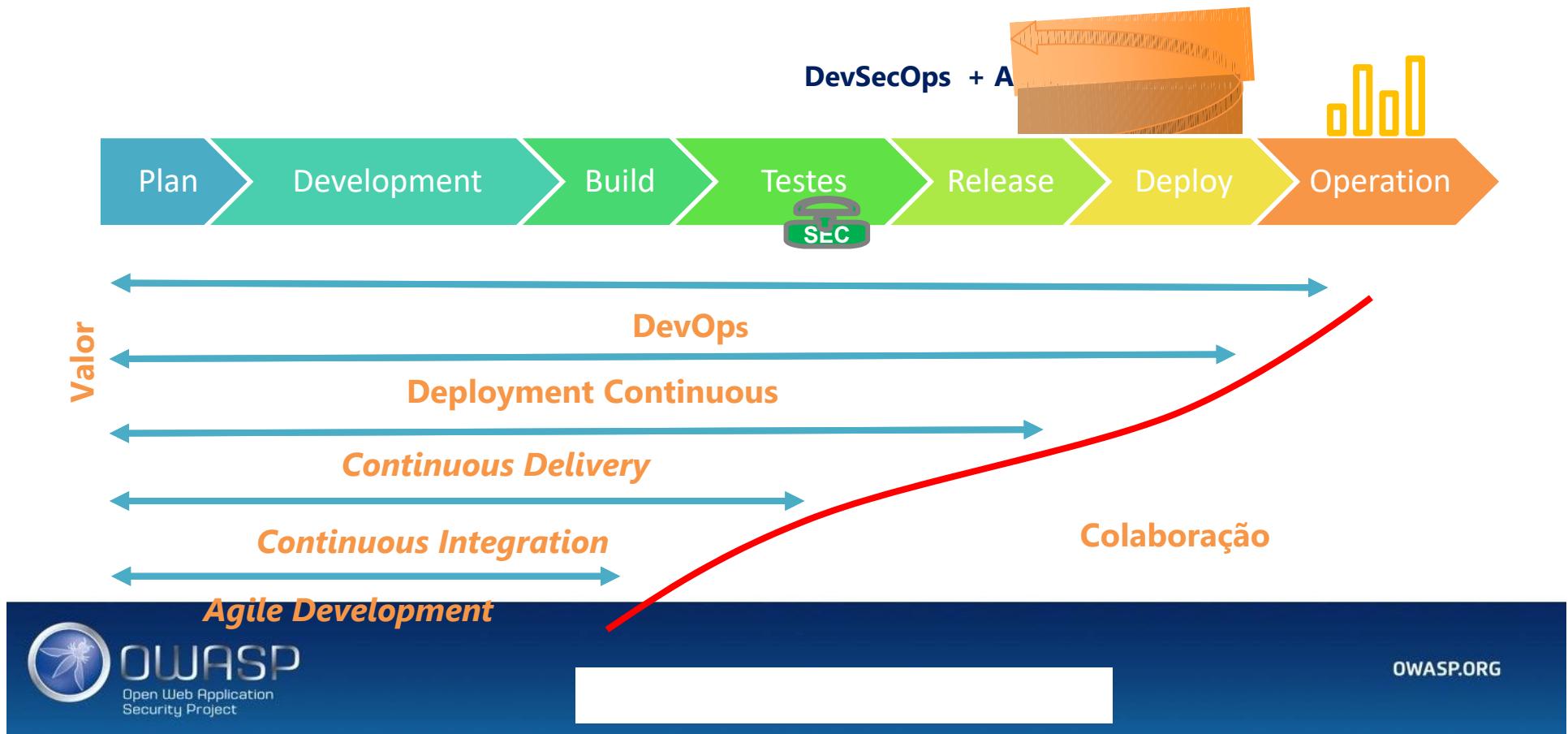


Processos e Ferramentas

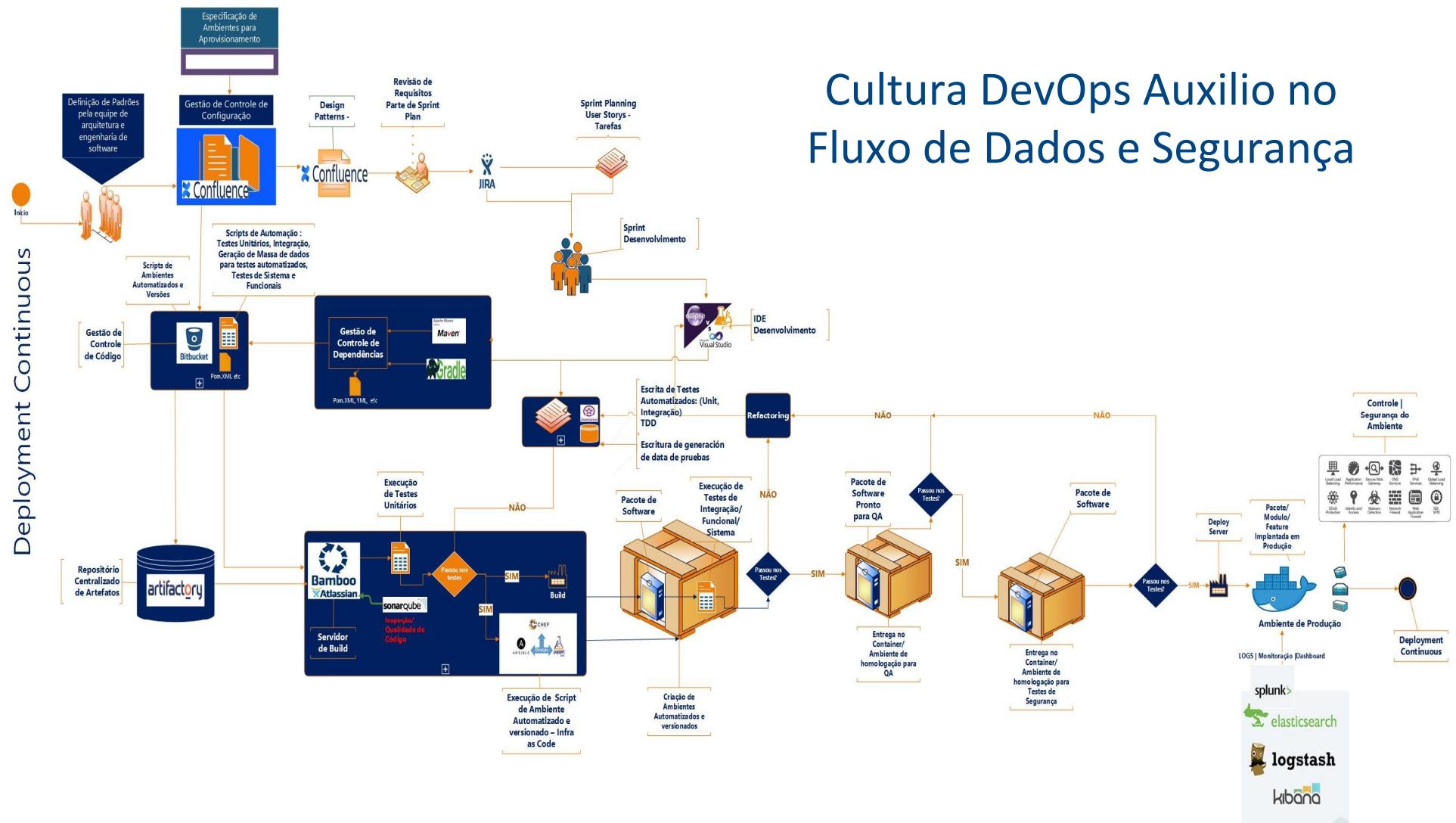
DevSecOps vs Outros Modelos



Cultura DevOps Auxilio no Fluxo de Dados e Segurança



Cultura DevOps Auxilio no Fluxo de Dados e Segurança



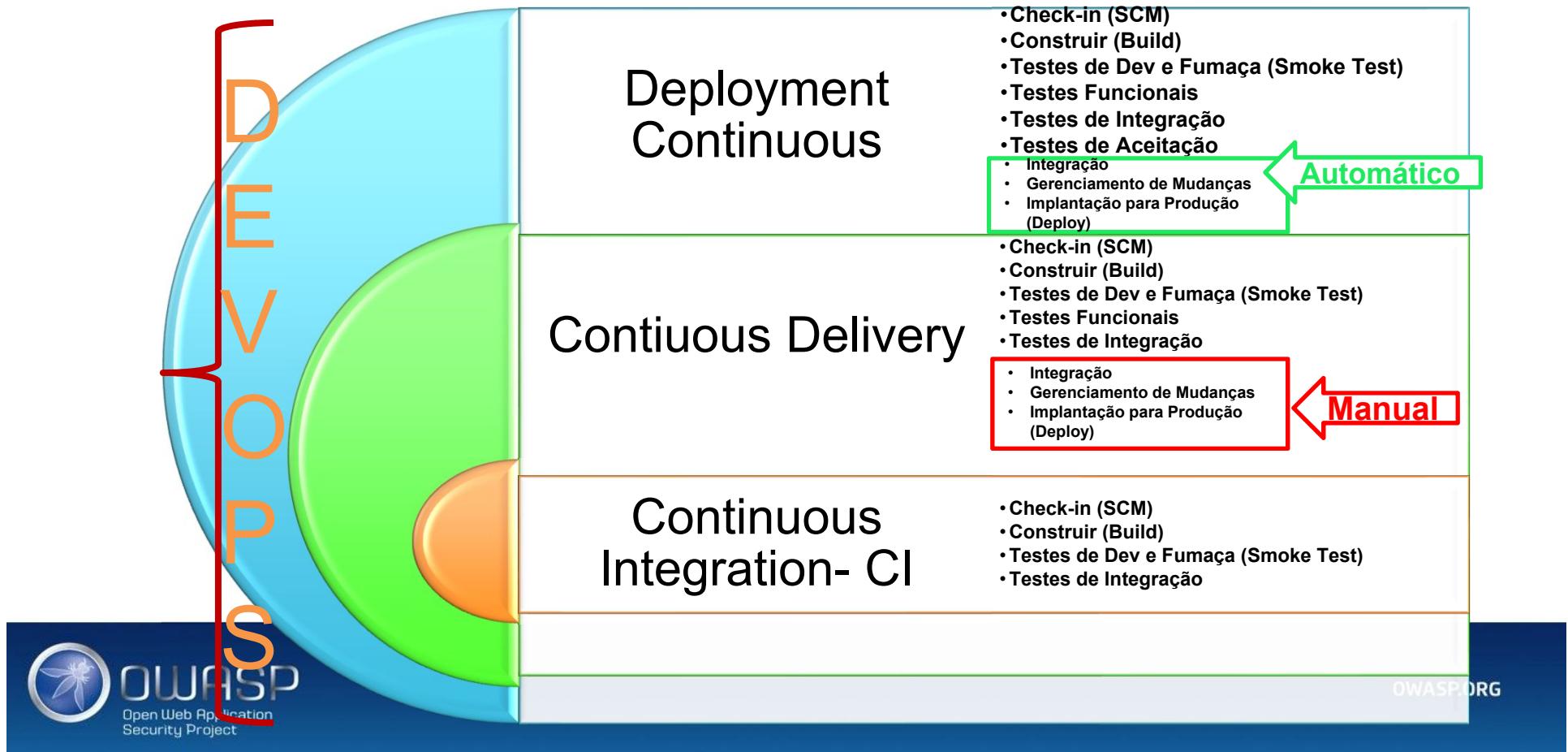
Cultura DevOps Auxilio no Fluxo de Dados e Segurança

Ganhos com Adoção de DevSecOps:



MITO VS REALIDADE

Diferença Entre DevOps X CI & CD:



MITO VS REALIDADE

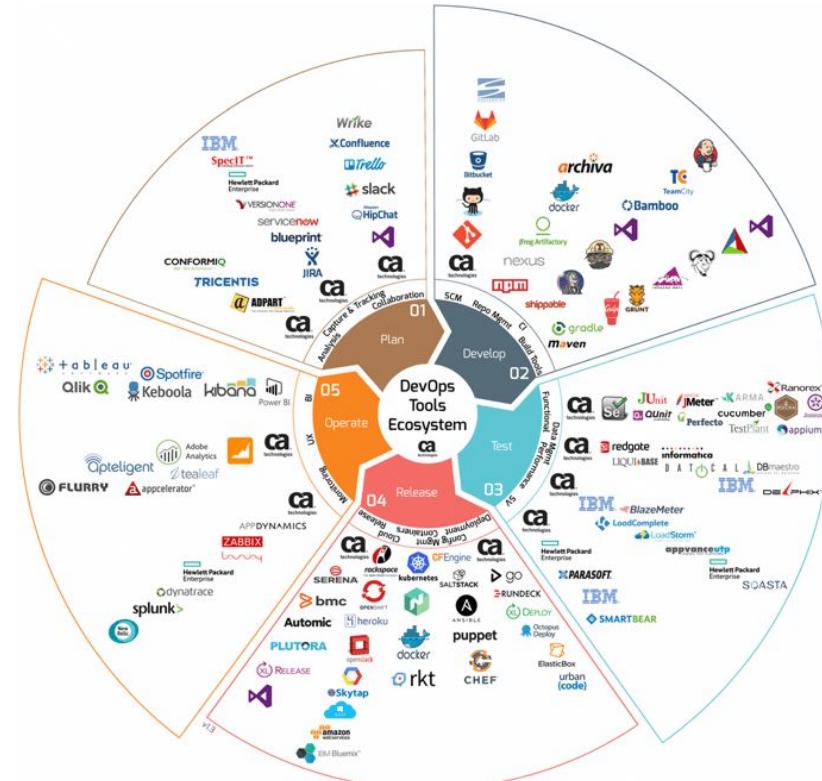
100% Automação

100% Seguro

100% Livre de Bugs

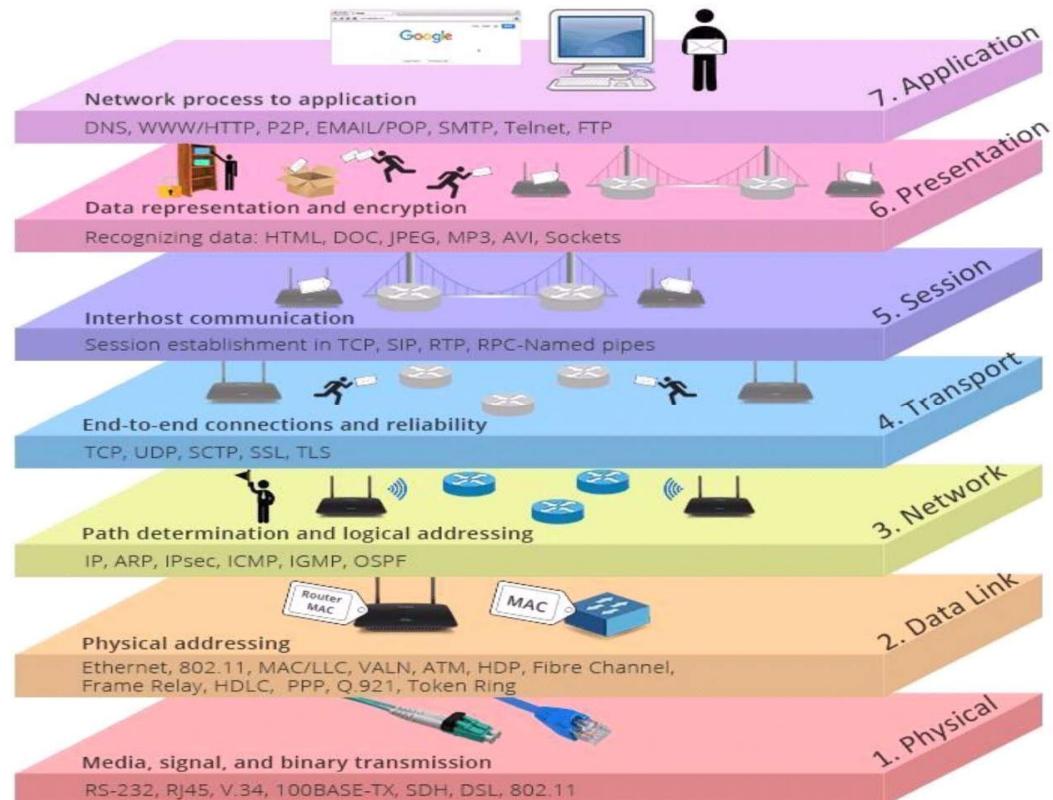
Bala de Prata

A Arte de Fazer o Triplo com a Metade das Pessoas





MITO VS REALIDADE



MITO VS REALIDADE

- 1: DevOps ou morra! – Fato
- 2: DevOps é desenvolvedores fazendo operações! – Mito
- 3: Projetos estão mortos – Mito
- 4: O DevOps não funciona em ambientes complexos! – Mito
- 5: é difícil “vender” o DevOps ao negócio – Mito
- 6: Agile é para engenheiros preguiçosos – Mito
- 7: se você não consegue codificar, não tem chance no DevOps – fato
- 8: os gestores desaparecem – mito
- 9: DevOps é somente para Descolados (Hipsters)



Compliance e Documentação de Testes

O que de fato importa para registrar?

Check- List de Requisitos Não Funcionais;

Modelagem de Ameaças;

Scripts para cenários de testes que podem ser automatizados com geração de relatórios

Definição das Ferramentas que serão utilizadas para execução de testes específicos de segurança conforme o domínio da aplicação e ameaças modeladas, que possibilitem a geração de relatórios

Compliance e Documentação de Testes

Plano de Testes, Casos/ Cenários de Testes/ Scripts de Testes/ Relatórios de Testes

Mapeamento da Aplicação - Projeto Exemplo



OWASP.ORG

Compliance e Documentação de Testes

Matriz de Rastreabilidade

Ameaças	Técnicas	Objetivos de Teste	Status	Observação
INJEÇÃO	Malicious File Execution	Em campos de Upload, verifique se a aplicação aceita arquivos não permitidos. Ex: XML, html e etc.	Não Executado	
	SQL Injection - Manual	Em campos de Pesquisa, execute testes manuais de SQL Injection.	Não Executado	
	SQL Injection - Automatizado	Execute a ferramenta SQL Inject Me na aplicação.	Não Executado	
	Fuzz Automatizado	Execute a ferramenta de pentest SQL Map na aplicação.	Não Executado	
	Fuzz Manual	Em campos de Texto e Pesquisa, execute testes manuais fuzzing na aplicação.	Não Executado	
QUEBRA DE AUTENTICAÇÃO E GERENCIAMENTO DE SESSÃO	Bypass vertical access	Usuários de privilégios diferentes, verifique se os usuários podem acessar dados para os quais o seu tipo de usuário tem permissão. (Acesso Vertical)	Não Executado	
	Bypass horizontal access	Usuários de privilégios diferentes, verifique se os usuários podem acessar modificar dados para os quais o seu tipo de usuário tem permissão. (Acesso Horizontal)	Não Executado	
	Password reset	Tela de login, verifique se o número de tentativas de login na aplicação é ilimitado.	Não Executado	
	Logout Application	Aplicação com "Logout", verificar se é possível acessar a aplicação ao clicar no botão "Voltar".	Não Executado	
	Renew cookies	Páginas que usam cookies, verifique se é possível renovar os cookies de sessão.	Não Executado	
ROSS SITE SCRIPTING - XSS	Xss - Text field	Em campos de Texto e Pesquisa, execute scripts de XSS.	Não Executado	
	Xss - URL	Na URL da aplicação, execute scripts de XSS.	Não Executado	
PROTEÇÃO DE ATAQUES INSUFICIENTES	Insecure Direct Object References	Validar código de erro não tratados nas páginas. (Usar Owasp Zap no modo Spider e Active Scan) Na URL da aplicação, verifique se é possível obter acesso através de Ids e funcionalidades. Execute a ferramenta de pentest (Nikto).	Não Executado	
	Forbidden url	Verifique na aplicação se é possível acessar url "proibidas" com referências expostas para atacantes (arquivo, código, diretório e etc.)	Não Executado	
	Autocomplete	Verifique se o autocomplete para campo de texto está habilitado.	Não Executado	
FALTA DE CONFIGURAÇÃO DE SEGURANÇA	Entry application points	Verifique se é possível extrair informações com os códigos de status HTTP da aplicação.	Não Executado	
	Login	Verifique se combinações diferentes na tentativa de login disponibilizam informações para o atacante.	Não Executado	
EXPOSIÇÃO DE DADOS SENSÍVEIS	Sensitive Data	Validar se aplicação utiliza criptografia e verificar o certificado.	Não Executado	
REQUISIÇÃO REMOTA FORJADA (CSRF)	Cross-Site Request Forgery (CSRF)	Execução da ferramenta Owasp Zap para validar todas as requisições na aplicação para evitar requisições falsas. Observe também se na aplicação é possível obter informações utilizando engenharia social.	Não Executado	
COMPONENTES COM VULNERABILIDADES CONHECIDAS	Components with Known Vulnerabilities	Verifique se os componentes utilizados possuem vulnerabilidades conhecidas expostas e já utilizadas, um ataque pode causar sérias perdas de dados ou o comprometimento do servidor.	Não Executado	
API's SUBPROTEGIDAS	UNDERPROTECTED API	UTILIZAÇÃO DE API'S SEM CRIPTOGRAFIA, FALTA DE FILA E MAPEAMENTO DE MICROSERVIÇOS, PODENDO CAUSAR PERDAS OU TROCA DE INFORMAÇÕES		
	Unvalidated redirects	Validar todos redirecionamentos que encaminham usuários para outras página na aplicação e fora da aplicação para determinar a página de destino.	Não Executado	
QUEBRA DE CONTROLE DE ACESSO	Missing Function Level Access Control	Falta de níveis de controle de Acesso, níveis de permissões e controle de acesso não definidos, permitindo a escalada de privilégios, comprometendo a integridade das informações.	Bloqueado	

Compliance e Documentação de Testes

Scripts de Testes

```
package com.example.tests;

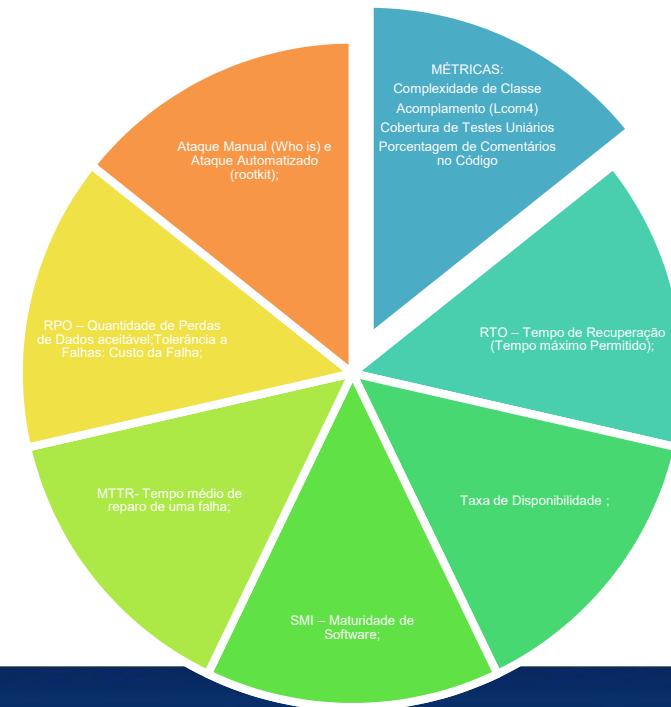
public class UntitledTestCase {
    private Selenium selenium;

    @Before public void setUp() throws Exception { WebDriver driver = new FirefoxDriver(); String baseUrl = "https://www.katalon.com/"; selenium = new WebDriverBackedSelenium(driver, baseUrl); }
    @Test
    public void testUntitledTestCase() throws Exception {
        selenium.open("https://twitter.com/");
        selenium.type("name=session[username_or_email]", "@Xxx_Xx");
        selenium.click("id=save_password");
        selenium.click("id=user-dropdown-toggle"); selenium.click("xpath=(//*[normalize-space(text()) and normalize-space(.)='Teclas de atalho'])[1]/following::button[1]");
        selenium.close();
    }
    @After
    public void tearDown() throws Exception { selenium.stop(); }
```

```
curl -X POST "https://api-hml.gs1br.org/oauth/access-token" -H "accept: application/json" -H "Authorization: Basic NWU1ODYxM2UtMmU5YS0zOGVmLWJkNjktNjJkYzU0OTNhZDIwOjhINGQ2ZTQ4LTUwMW MtM2VkJC05Njgx LTUwYmYyZDBjYjQ5Mw==" -H "Content-Type: application/json" -d "{ \"grant_type\": \"password\", \"username\": \"123abc@gmail.com\", \"password\": \"G$@2018\", \"cpfCnpj\": \"007007160218\"}"
```

Como encaixar OWASP TOP 10 , Governança de Dados e Compliance? Métricas

- CONTROLE DE HISTÓRICO DE TESTES EXECUTADOS COM FALHA;
- CONTROLE DE RELATÓRIOS DE TESTES EXECUTADOS ;
- EXTRAÇÃO DE MÉTRICAS APARTIR DOS RESULTADOS DO TESTES EXECUTADOS;
- MONITORAMENTO DO CICLO VIDA DA APLICAÇÃO VS EVOLUÇÃO DE PROBLEMAS;
- HISTOGRAMA DE FALHAS;
- CONTROLE DE MUDANÇAS VS RELATÓRIO DE FALHAS PÓS MUDANÇAS;
- GESTÃO DE CONFIGURAÇÃO ATUALIZADA CONFORME CÍCLO DE VIDA DA APLICAÇÃO



Como encaixar OWASP TOP 10 , Governança de Dados e Compliance?

Técnicas, Ferramentas e Testes que o QA pode Utilizar:

Fuzzy utilizando Postman no Teste de API

SQL Injection no testes de Integração (Owasp top 10)

XSSInjection (Xenotic Exploit OWASP)

Teste de Acesso ao Aplicativo (web ou mobile) - estes detalhados de todas as funções e direitos devem ser realizados. O testador deve criar várias contas de usuário com funções diferentes e múltiplas – Verificação de Autenticação e Autorização – Gestão de Acesso e Identidade; Verificar de Proteção de Dados - O testador deve verificar se, quando a informação está sendo transmitida entre o cliente e o servidor, ela não é exibida na barra de endereços de um navegador da Web em um formato comprehensível. Se alguma dessas verificações falhar, o aplicativo definitivamente terá uma falha de segurança – Verificar Protocolo e Criptografia;



OWASP
Open Web Application
Security Project

Referências:

- COSTA, I; NETO, M; COSTA NETO, P; JUNIOR, J. et al. Qualidade em Tecnologia da Informação. São Paulo: Editora Atlas, 2013. CORREIA, M. Segurança no Software. Lisboa: Editora, 2010. LYRA, M. et al. Segurança e Auditoria em Sistemas de Informação. Rio de Janeiro: Editora Ciência Moderna, 2008. MIGUEL, A. Gestão de Projectos de Software. Lisboa: Editora QFCA, 2010. RIOS, E; MOREIRA, T. et al. Teste de Software. Rio de Janeiro: Alta Books, 2013. C. 2010. SOLOMON, M.G; KIM, D. et at. Fundamentos de Segurança de Sistemas de Informação. Rio de Janeiro: Editora LTC, 2014. PADUA, W. A. et al. Engenharia de Software Fundamentos, Métodos e Padrões. Rio de Janeiro: Editora LTC. MORAIS, Gleicon - "CAIXA DE FERRAMENTAS DEVOPS – Casa do Código, 2017 São Paulo, SP.
- Exin White Paper
 - <https://cipher.com.br/2018/03/13/diferenca-entre-analise-de-vulnerabilidade-e-teste-de-penetracao/>
 - <https://pt.slideshare.net/JulianoPadilha1/engenharia-de-software-ii-teste-de-seguranca-de-software>
 - <https://blog.conviso.com.br/tag/testes-de-seguranca/>
 - <https://www.cogniti.com/security-testing/>
 - https://pt.wikipedia.org/wiki/Ficheiro:%D9%83%D8%A7%D9%84%D9%8A_%D9%84%D9%8A%D9%86%D9%83%D8%B3.png
 - https://pt.wikipedia.org/wiki/Teste_de_seguran%C3%A7a
 - https://pt.wikipedia.org/wiki/Kali_Linux
 - https://www.google.com.br/search?q=security+testing+tools&tbs=isch&tbs=rimg:CTx_1PvjxCHzRIjhfkB-2XWXgUKD20_1Izv8jr7NSG4abVXX_1THw6ngP-DmVUPRj1kq5tQcAjQ-M9CczfWFb1U3QVkJSoSCV-GQH7ZdZeBEUHQvxzS0fU1
 - https://www.owasp.org/index.php/OWASP_Testing_Guide_Appendix_C:_Fuzz_Vectors
 - <https://www.owasp.org/images/1/19/OTGv4.pdf>
 - https://www.ibm.com/developerworks/community/blogs/a9ba1efe-b731-4317-9724-a181d6155e3a/entry/accelerating_maximo_development_with_continuous_delivery?lang=en
 - <https://www.gp4us.com.br/backlog-do-produto/>
 - <http://www.datacenterdynamics.com.br/focus/archive/2015/02/empresas-brasileiras-faturam-29-mais-ao-adotar-devops-diz-pesquisa-da-ca-techn>
 - <https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/beyond-agile-reorganizing-it-for-faster-software-delivery>
 - <https://pt.wikipedia.org/wiki/Compliance>



OWASP
Open Web Application
Security Project

Obrigado!

Para mais informações, acesse:

www.owasp.org

Alessandra Martins
monteiromartins@bol.com.br