NÃO PODE FALTAR

ANÁLISE DE VULNERABILIDADE E *PENTEST*

Emilio Tissato Nakamura

O QUE É UM TESTE DE SEGURANÇA?

Um teste de segurança é um processo de análises e avaliações de riscos e análises de vulnerabilidades. É o processo de comparar o estado de um sistema ou aplicação de acordo com um conjunto de critérios.

Fonte: Shutterstock.

Deseja ouvir este material?

Áudio disponível no material digital.

PRATICAR PARA APRENDER

Olá, nesta seção vamos estudar o elemento do risco que é explorado pelos agentes de ameaça nos ataques: a vulnerabilidade. O ambiente das empresas é formado por sistemas, plataformas, aplicações e aplicativos que são utilizados por colaboradores e clientes para acessar diferentes informações e serviços. Há um conjunto de ativos que formam este ambiente e qualquer um destes ativos pode ser atacado e resultar em incidentes de segurança e impactos para a empresa.

Os testes de segurança são importantes para a gestão de segurança da informação porque identificam as vulnerabilidades, que podem ser tratadas. E há diferentes formas de fazer testes de segurança e identificar as vulnerabilidades. O objetivo é a sua empresa disponibilizar serviços seguros, sem vulnerabilidades. Sem os testes de segurança, a sua empresa pode estar expondo informações sigilosas e a privacidade de clientes, colaboradores e parceiros.

Se a sua empresa desenvolve *software*, ela deve disponibilizar o sistema de uma forma segura, seguindo práticas que vão eliminando as vulnerabilidades desde o início do desenvolvimento até o período após a implantação em ambiente de produção.

Se a sua empresa utiliza *software* de terceiros, ela deve fazer testes de segurança para garantir que o ambiente da empresa, composta por softwares de diferentes fornecedores e de naturezas diferentes, esteja seguro.

E os testes de segurança são uma das principais atividades de empresas especializadas em segurança e privacidade, com a oferta de serviços de análise de vulnerabilidades e pentests, por exemplo.

Você é o especialista em segurança e privacidade de um inovador *site* de comércio *online* em que pequenos negócios são conectados com os consumidores em uma plataforma digital baseada no uso de inteligência artificial. A sua função é essencial para a empresa, e você participa de todas as decisões sobre a evolução da plataforma. Há as questões

envolvidas com o desenvolvimento seguro, para que vulnerabilidades não sejam inseridas. Há ainda as questões de segurança e privacidade envolvidas com o uso de provedor de nuvem. E, como a empresa trabalha com inteligência artificial, há necessidade de fazer o desenvolvimento utilizando bases de dados que não interfiram na privacidade dos clientes.

Além da segurança da informação da plataforma da empresa, que está hospedada em um provedor em nuvem na Europa, você tem três preocupações principais:

1. Como diminuir as possíveis fraudes cometidas por usuários falsos que se passam por clientes, com uso de identidades falsas ou uso de recursos financeiros ilícitos.
2. Como diminuir as possíveis fraudes cometidas por pequenos negócios falsos, que podem não cumprir os compromissos comerciais estabelecidos com os clientes que utilizam a plataforma digital.
3. Como proteger os dados pessoais dos clientes principalmente contra vazamentos, que pode levar a sanções previstas na LGPD.

Após já ter mostrado um planejamento sobre os aspectos que devem ser considerados pela empresa para a definição de uma estratégia de segurança e privacidade, com o seu direcionamento quanto à segurança em transações web e para a plataforma móvel, agora vamos para o próximo passo.

O que deve ser planejado agora é a forma como a empresa deve tratar as vulnerabilidades, tanto da plataforma *web* quanto da plataforma móvel. Mostre as perspectivas envolvidas com a gestão de vulnerabilidades e a razão de precisarem ser tratadas antes das plataformas irem para o ambiente de produção.

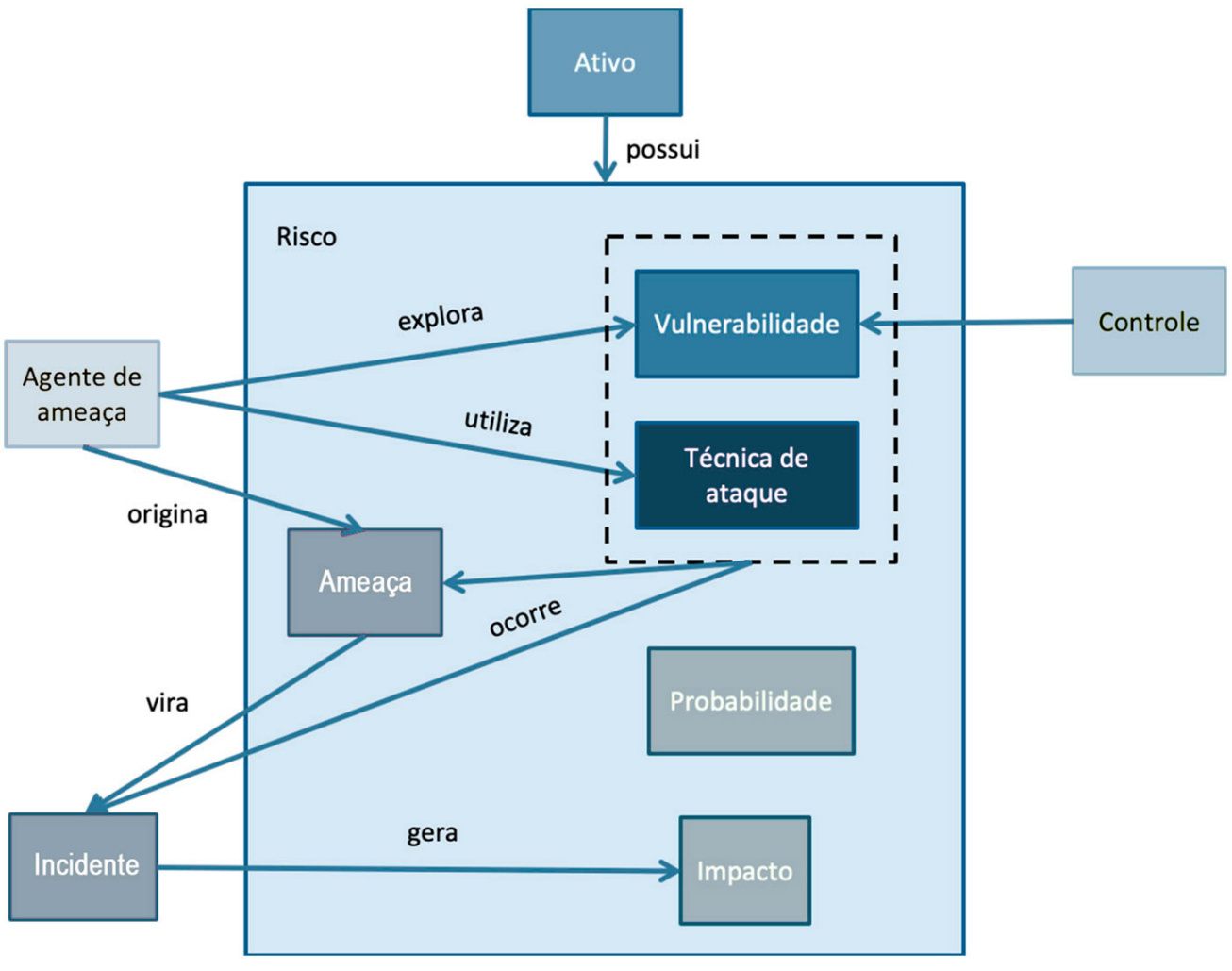
Continuando com o foco nas vulnerabilidades, mostre o que a empresa fará durante as fases do desenvolvimento das plataformas e o que será feito após a implantação.

Nesta aula, você verá que há diversas possibilidades de realização dos testes de segurança, que são baseados na origem dos testes e no nível de conhecimento prévio do ambiente em análise. Você verá que a qualificação e as qualidades técnicas do profissional são essenciais para que bons resultados sejam alcançados. Algumas metodologias estabelecem o que deve ser feito em cada passo dos testes de segurança e são importantes para você aplicar na sua empresa, seja como profissional de segurança ou como consultor.

CONCEITO-CHAVE

Uma das principais atividades dos profissionais de segurança da informação são as atividades relacionadas às vulnerabilidades dos ativos. Quando um ataque ou incidente de segurança acontece, ele é resultado da exploração de vulnerabilidades de ativos por um agente de ameaça (Figura 3.15).

Figura 3.15 | Ativos possuem vulnerabilidades que podem ser exploradas



Fonte: elaborada pelo autor.

O risco é a probabilidade de um agente de ameaça explorar vulnerabilidades de ativos utilizando alguma técnica de ataque, o que faz com que uma ameaça se torne um incidente de segurança,

causando impactos à empresa. Uma das principais formas de reduzir os riscos das empresas é o tratamento das vulnerabilidades, para que elas não possam ser exploradas (NAKAMURA, 2016).

GESTÃO DE VULNERABILIDADES

As vulnerabilidades têm natureza complexa, já que são descobertas o tempo todo, surgem e são criadas em uma velocidade ainda maior. Com os ambientes mudando o tempo todo e com uso e integração de diferentes tecnologias, há sempre novas vulnerabilidades dos novos ativos das empresas. Assim, é importante que as vulnerabilidades sejam tratadas por um processo de gestão de vulnerabilidades que organiza as ações para a descoberta das inúmeras vulnerabilidades em diferentes ativos, levando em consideração a dinâmica das novas vulnerabilidades e dos ativos das empresas, que estão em constante mudança.

A gestão de vulnerabilidades engloba alguns processos que podem ser vistos na Figura 3.16 (CAVALANCIA, 2020):

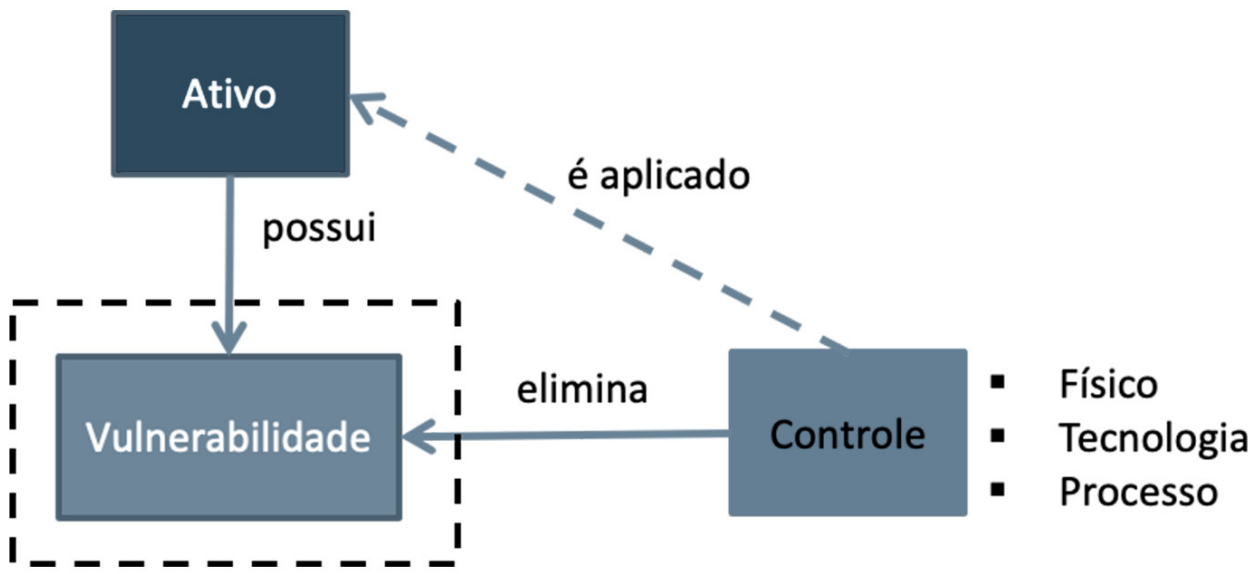
- **Descoberta:** não é possível proteger o que não se conhece e este processo envolve o inventário de ativos, incluindo sistema operacional, serviços, aplicações e configurações.
- **Priorização de ativos:** os ativos descobertos precisar ser priorizados de acordo com uma visão de riscos para a empresa.
- **Avaliação:** estabelece uma linha de base para as vulnerabilidades e os riscos.
- **Relatório:** provê a visibilidade para todos da empresa, principalmente os executivos, que precisam entender o estado atual dos riscos e vulnerabilidades.
- **Remediação:** as vulnerabilidades, de acordo com a avaliação, são remediadas com a aplicação de controles de segurança.
- **Verificação:** a remediação é verificada para confirmar que a vulnerabilidade foi tratada corretamente.

Figura 3.16 | Processos da gestão de vulnerabilidades

Fonte: adaptado de Cavalancia (2020).

A identificação de vulnerabilidades é o início dos trabalhos para proteger as empresas e pode ser feita de diferentes formas. Uma vez descoberta e validadas as vulnerabilidades, elas devem ser tratadas com os controles de segurança. Os controles de segurança a serem aplicados na remediação das vulnerabilidades podem ser dos tipos físico, tecnológico ou processual (Figura 3.17).

Figura 3.17 | Controles de segurança



Fonte: elaborada pelo autor.

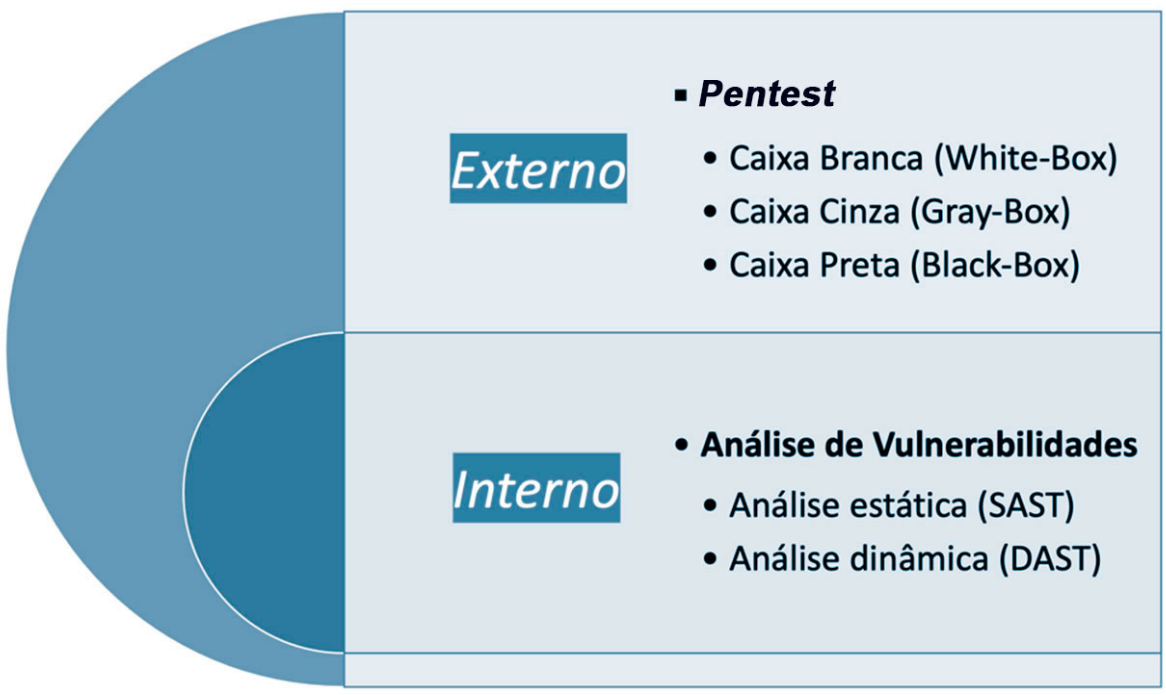
Ver anotações 0

TESTES DE SEGURANÇA

Há diferentes testes de segurança, como as análises e avaliações de riscos e as análises de vulnerabilidades, que focam tradicionalmente em aspectos tecnológicos. Para a *Open Web Application Security Project* (OWASP), que foca em aplicações *web*, teste de segurança é o processo de comparar o estado de um sistema ou aplicação de acordo com um conjunto de critérios (OWASP, 2014). Eles podem ser feitos no final do desenvolvimento ou fazer parte do ciclo de desenvolvimento desde o início, com a implementação de requisitos e testes de segurança automatizados (OWASP, 2019).

Os testes de segurança refletem diferentes visões, as quais envolvem variáveis como a origem dos testes (interno ou externo), as informações prévias disponíveis para os testes, o uso de ferramentas automatizadas e a qualificação dos profissionais. Estas variáveis são importantes, serão discutidas durante esta seção e indicam os tipos de teste de segurança envolvido, tais como os que podem ser vistos na Figura 3.18.

Figura 3.18 | Testes de segurança mais conhecidos



Fonte: elaborada pelo autor.

REFLITA

É preciso ter a mentalidade correta para os testes de segurança. Testes de segurança requerem um pensamento “fora da caixa”. Casos de uso normais irão testar o comportamento normal da aplicação em que o usuário está

Ver anotações

utilizando as funções da forma como é esperado. Em testes de segurança, é preciso ir além das expectativas tradicionais e ter um pensamento de atacante, ou seja, daquele que está tentando quebrar a aplicação. A criatividade pode ajudar a determinar o que um dado não esperado pode causar na aplicação. Há casos em que as premissas não são sempre verdadeiras e podem ser subvertidas. E essa mentalidade faz com que os testes de segurança sejam feitos de acordo com critérios mentais que não são bem definidos ou completos, o que leva os demais a tratarem os testes de segurança como uma arte do mal. O papel do profissional é ainda mais valioso pelo fato das ferramentas automatizadas não terem criatividade, apenas implementando frameworks comuns (OWASP, 2014).

Um teste de segurança típico é estruturado tipicamente de acordo com as seguintes fases (OWASP, 2019), como pode ser visto na Figura 3.19:

- **Preparação:** definição do escopo do teste de segurança, incluindo a identificação dos controles de segurança aplicáveis, objetivos organizacionais do teste, e os dados sensíveis. Envolve ainda o alinhamento com o cliente e as medidas legais para os testes, que incluem a autorização.
- **Obtenção de informações:** análise do ambiente e da arquitetura da aplicação para o entendimento do contexto.
- **Mapeamento:** as informações obtidas até esta fase podem ser complementadas pelo uso de ferramentas automatizadas ou exploração manual. O mapeamento provê o entendimento da aplicação, os pontos de entrada, os dados e as potenciais vulnerabilidades. O mapeamento também inclui a criação dos casos de testes que serão executados, sendo o modelo de ameaças um dos artefatos que podem ser utilizados.

- **Exploração:** o profissional tenta atacar a aplicação explorando as vulnerabilidades identificadas durante a fase anterior, com a intenção de validá-las. Alguns parâmetros a serem considerados são o potencial de dano, a facilidade de reprodução do ataque, a facilidade de executar o ataque, os usuários afetados e a facilidade de descobrir a vulnerabilidade.
- **Relatório:** o profissional relaciona as vulnerabilidades que puderam ser exploradas, incluindo o escopo do comprometimento.

Figura 3.19 | Teste de segurança típico

Fonte: adaptada de OWASP (2019).

REFLITA

Fazer um teste de segurança superficial e considerá-lo completo é tão crítico quanto não fazê-lo, pela falsa sensação de segurança gerada. É importante que as vulnerabilidades encontradas sejam validadas, já que falsos negativos (falhas não encontradas) são fatais e falsos positivos (falhas apontadas que não existem) tiram a credibilidade dos resultados como um todo. Toda a lógica da aplicação deve ser testada e todo cenário de caso de uso deve ser analisado em busca de possíveis vulnerabilidades (OWASP, 2014).

ANÁLISE DE VULNERABILIDADES

A análise de vulnerabilidades compreende a busca por vulnerabilidades nos ativos de uma forma manual ou com o uso de ferramentas automatizadas, como os *scanners*. Os tipos de análise de vulnerabilidades são as análises estática e dinâmica (KOUSSA, 2018) (OWASP, 2019).

A análise estática, ou *Static Application Security Testing* (SAST), envolve a análise dos componentes do sistema sem a sua execução, pela análise manual ou automatizada do código-fonte. A análise manual exige proficiência na linguagem e no framework usado pela aplicação e possibilita a identificação de vulnerabilidades na lógica de negócios, violações de padrões e falhas na especificação, especialmente quando o código é tecnicamente seguro, mas com falhas na lógica, que são difíceis de serem detectados por ferramentas automatizadas. Já a análise automatizada é feita com ferramentas que checam o código-fonte por conformidade com um conjunto pré-definido de regras ou melhores práticas da indústria (OWASP, 2019).

> **EXEMPLIFICANDO**
>
> A revisão manual do código pode ser feita com o uso de métodos mais básicos de busca de palavras-chave no código-fonte, ou com a análise linha a linha do código-fonte. Também podem ser utilizados os ambientes de desenvolvimento, ou *Integrated Development Environments* (IDEs) (OWASP, 2019).

A análise dinâmica, ou *Dynamic Application Security Testing* (DAST), envolve a análise do sistema durante a sua execução, em tempo real, de forma manual ou automatizada. Normalmente, a análise dinâmica não provê as informações que a análise estática provê, mas detecta elementos sob o ponto de vista do usuário, como os ativos, funções, pontos de entrada e outros. A análise dinâmica é conduzida na camada da plataforma e nos serviços e *Application Programming Interfaces*

(APIs) do *backend*, que são locais em que as requisições e respostas das aplicações podem ser analisadas. Os resultados são referentes, principalmente, a problemas de confidencialidade no trânsito, de autenticação e autorização, além de erros de configuração do servidor (OWASP, 2019).

O SAST e DAST podem ser adotados pelas próprias equipes de desenvolvimento no contexto do DevSecOp, que é um conceito importante que pode ser seguido para o desenvolvimento de *software*, ao integrar os testes de segurança na esteira de desenvolvimento, envolvendo a integração contínua e a entrega contínua (CONSTANTIN, 2020).

REFLITA

Tratar falsos positivos ou alarmes falsos gerados por ferramentas automatizadas é fundamental. Um exemplo é uma vulnerabilidade em um servidor *backend*, que pode ser explorada a partir de um navegador *web*, mas não a partir de um aplicativo móvel. Isso ocorre em caso de ataques de *Cross-site Request Forgery* (CSRF) e *Cross-Site Scripting* (XSS) (OWASP, 2019).

ASSIMILE

SAST deve ser aplicado no código-fonte e é importante para remover as vulnerabilidades do código antes de o *software* entrar em produção. O DAST também deve ser realizado antes de o *software* entrar em produção e o teste é com o *software* funcionando, testando-se as interfaces existentes. Há ainda um teste de segurança conhecido como IAST (*Interactive Application Security Testing*), que faz os testes de segurança de uma forma interativa, combinando os testes estáticos e dinâmicos (SAST e DAST).

REFLITA

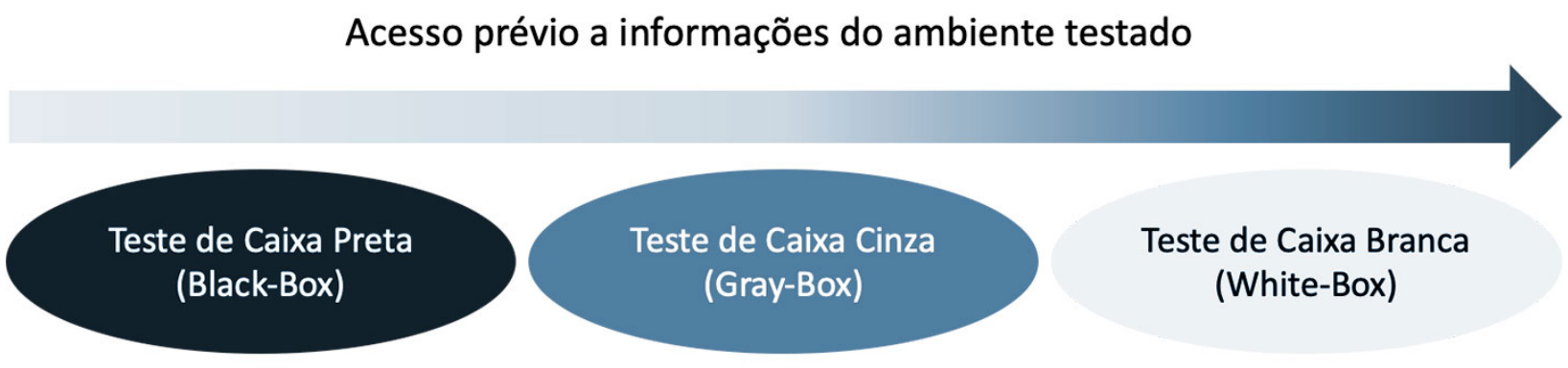
As ferramentas de análise automatizada do SAST geram os resultados com avisos e alertas das violações detectadas e podem funcionar como *plug-ins* nos IDEs. Os resultados dessas ferramentas podem gerar falsos positivos, principalmente se não forem configuradas para o ambiente específico do sistema. Assim, é fundamental que os resultados automatizados sejam sempre revisados por um profissional de segurança (OWASP, 2019).

PENTEST

Os testes de penetração ou *pentests* são também conhecidos como testes de intrusão e *ethical hacking* e são realizados a partir do ambiente externo. Os objetivos são determinar “se” e “como” um agente de ameaça pode obter um acesso não autorizado a ativos que afetam um ambiente, e confirmar se os controles requeridos por padrão, regulamento ou legislação estão implementados. Envolve ainda identificar meios de explorar vulnerabilidades para driblar os controles de segurança dos componentes do sistema (PCI, 2017).

Há três tipos de *pentests*, que depende das informações do ambiente obtidas antes dos testes de segurança, como pode ser visto na Figura 3.20.

Figura 3.20 | Tipos de *pentest*



Fonte: adaptada de OWASP (2019), PCI (2017).

O **teste de caixa preta (*Black-Box)*** é também conhecido como teste com conhecimento zero, já que é conduzido sem qualquer informação sobre o ambiente que está sendo testado. O objetivo é que o

profissional faça o teste como se fosse um atacante real, explorando o uso de informações públicas e que podem ser obtidas sem restrição por qualquer atacante (OWASP, 2019).

> **REFLITA**
>
> Os resultados dos testes de caixa preta podem impressionar e serem úteis para demonstrar como as vulnerabilidades são exploradas em um ambiente de produção. Porém, não são muito efetivos ou eficientes em tornar a aplicação mais segura. Há dificuldades em testes dinâmicos que exploram todo o código, particularmente quando há uma série de condições interligadas. Assim, se o código-fonte da aplicação estiver disponível, é importante que ele seja disponibilizado para o profissional de segurança para que este possa utilizá-lo no teste (OWASP, 2014).

O **teste de caixa branca (*White-Box*)** é também conhecido como teste com conhecimento total e é conduzido com todo o conhecimento sobre o ambiente, que engloba código-fonte, documentações e diagramas. Este tipo de teste é mais rápido do que o teste de caixa preta, porque há a transparência e o conhecimento permite a construção de casos de teste mais sofisticados e granulares (OWASP, 2019).

> **REFLITA**
>
> O acesso ao código-fonte no teste de caixa branca (*White-Box*) não simula ataques externos, mas simplifica a identificação de vulnerabilidades, ao possibilitar a identificação de anomalias ou comportamentos suspeitos diretamente no código. A decompilação de aplicações pode ser utilizada no teste de caixa preta (*Black-Box*), porém o código-fonte pode estar ofuscado e revertê-lo pode ser trabalhoso (OWASP, 2019).

Já o **teste de caixa cinza (*Gray-Box*)** é o teste em que alguma informação é provida para o profissional, como uma credencial de acesso, enquanto outras informações têm de ser descobertas. Este teste é bastante comum, devido aos custos, tempo de execução e escopo do teste (OWASP, 2019).

> **DICA**
>
> *Hackathon* é um evento que reúne programadores, *designers* e outros profissionais ligados ao desenvolvimento de *software* em maratonas de trabalho com o objetivo de criar soluções específicas para um ou vários desafios (GOMES, 2017). Ele pode envolver aspectos de segurança da informação e há, ainda, o *Capture The Flag* (CTF), que é uma modalidade de competição voltada a desvendar problemas sobre segurança da informação, avaliando, de forma gamificada, habilidades como vulnerabilidade da rede, criptografia e programação, entre outras (MENA, 2018). E há testes públicos de segurança, como o teste público de segurança promovido pelo TSE, visando um conjunto de ações controladas a fim de identificar vulnerabilidades e falhas relacionadas à violação da integridade ou do sigilo do voto em uma eleição, com a apresentação de sugestões de melhoria de componentes do sistema eletrônico (TSE, 2017).
>
> As empresas também podem criar programas de *Bug Bounty*, em que oferecem prêmios para quem encontrar falhas em seus sistemas, que variam de acordo com o nível de gravidade de cada vulnerabilidade encontrada (WARBURTON, 2020).

METODOLOGIA OWASP TESTING PROJECT

A OWASP Testing Project foca em aplicações web e visa a construção de aplicações mais confiáveis e seguras. A metodologia segue as premissas de que a prática de testar o *software* deve estar em todo o ciclo de vida

de desenvolvimento do seu desenvolvimento (*Software Development Life Cycle*, SDLC) (Figura 3.21) e que uma das melhores maneiras de prevenir bugs de segurança em aplicações em produção é o SDLC incluir a segurança em cada uma de suas fases.

Figura 3.21 | Ciclo de vida de desenvolvimento de *software* (SDLC)

Fonte: adaptada de OWASP (2014).

A metodologia de testes do OWASP compreende técnicas e atividades para cada fase do SDLC, como pode ser vista na Figura 3.22. Elas são voltadas para serem aplicadas nas empresas que desenvolvem *software*, e trata dos seguintes pontos (OWASP, 2014):

- **Fase 1 - Antes de o desenvolvimento iniciar**
  - **Fase 1.1 - Definição do SDLC**: define em qual estágio a segurança é inerente no processo de desenvolvimento.
  - **Fase 1.2 - Revisão de políticas e padrões**: assegura que as equipes possam desenvolver as atividades de acordo com as políticas, padrões e documentações.
  - **Fase 1.3 - Desenvolvimento de métricas**: dá visibilidade ao processo e ao produto com as métricas a serem medidas.

- **Fase 2 - Durante a definição e especificação**
  - **Fase 2.1 - Revisão dos requisitos de segurança**: define como a aplicação deve funcionar na perspectiva de segurança. Os requisitos de segurança precisam ser testados. Os requisitos não devem ser ambíguos e incluem mecanismos como gerenciamento de usuários, autenticação, autorização, confidencialidade de dados, integridade, contabilidade, gerenciamento de sessão, segurança no transporte, segregação em camadas, conformidade com legislação e padrões.
  - **Fase 2.2 - Revisão da especificação e arquitetura**: testa artefatos como modelos, documentos textuais e outros documentos, para analisar os requisitos de segurança considerados.
  - **Fase 2.3 - Criação e revisão dos modelos UML**: confirma que o entendimento do funcionamento da aplicação é exato após a especificação e arquitetura e modelos *Unified Modeling Language* (UML).
  - **Fase 2.4 - Criação e revisão do modelo de ameaças**: utiliza cenários de ameaças realísticos sobre a especificação, arquitetura e modelos UML para a modelagem de ameaças. As ameaças devem ter sido mitigadas, aceitas pelo negócio ou transferidas para terceiros, como empresas de seguros. Caso não haja estratégias de mitigação, a especificação deve ser alterada.
- **Fase 3 - Durante o desenvolvimento**
  - **Fase 3.1 - Execução simulada do código**: executa (*walk through*) o código com os desenvolvedores e/ou arquitetos do sistema, em um processo alto nível de execução simulada do código em que os desenvolvedores podem explicar a lógica e o fluxo do código implementado. Esta fase permite o entendimento geral do código e possibilita que os desenvolvedores expliquem a

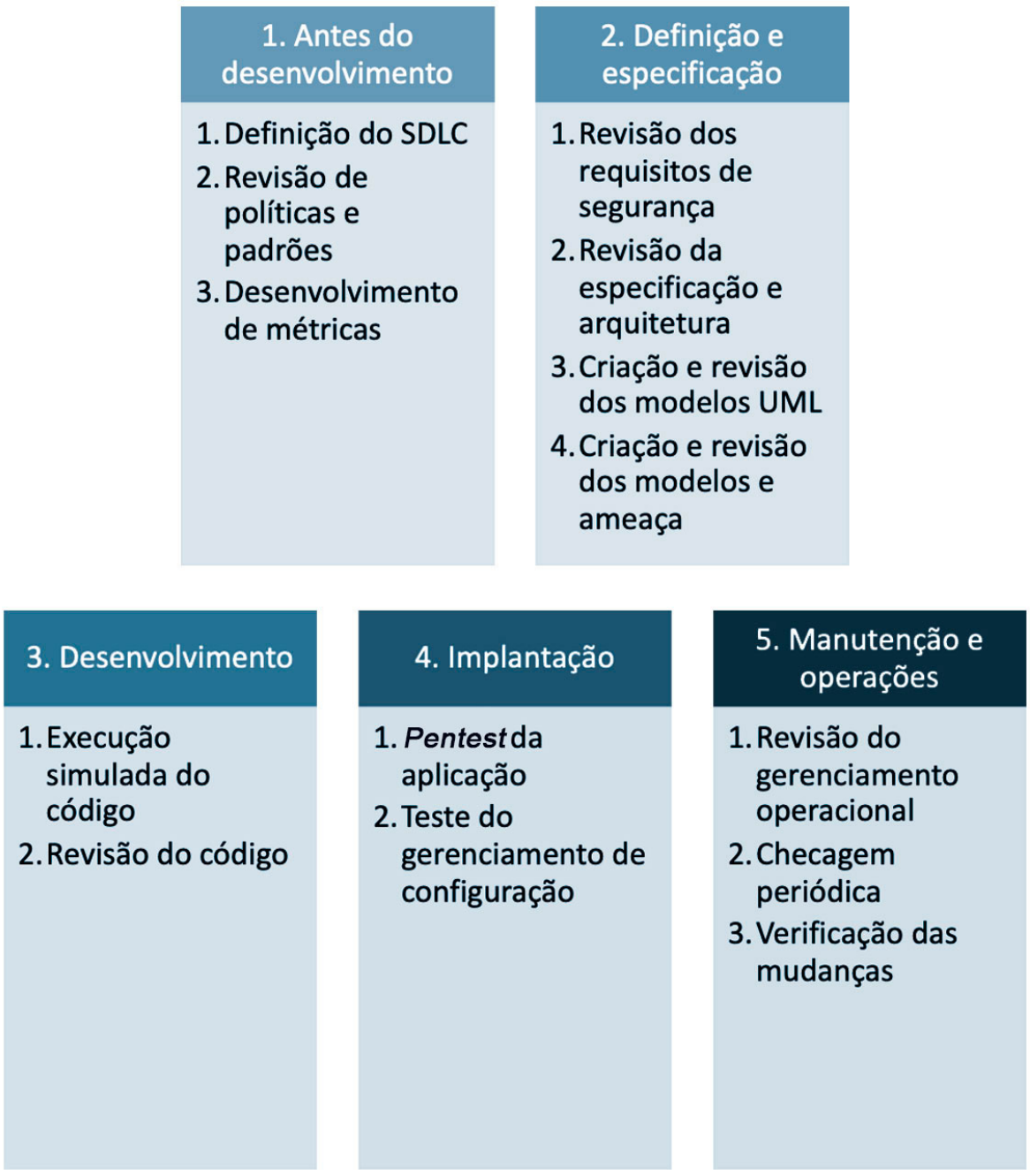
razão de alguns trechos específicos desenvolvidos. Não é objetivo revisar o código, mas ter um entendimento em alto nível do fluxo, do *layout* e da estrutura do código da aplicação.

- **Fase 3.2 - Revisão do código**: valida o código com a revisão estática, de acordo com alguns objetivos, que incluem os requisitos de negócios para disponibilidade, confidencialidade e integridade, exposição técnica de vulnerabilidades do OWASP Top 10, pontos específicos de linguagens ou *frameworks*, além de requisitos específicos de alguma indústria.

- **Fase 4 - Durante a implantação**
  - **Fase 4.1 - *Pentest* da aplicação**: provê uma última checagem após os testes dos requisitos, análise da especificação e revisão do código.
  - **Fase 4.2 - Teste do gerenciamento de configuração**: testa a forma como a infraestrutura é implantada e segura, principalmente quanto a instalações-padrão e vulneráveis.

- **Fase 5. Manutenção e operações**
  - **Fase 5.1 - Revisão do gerenciamento operacional**: analisa a forma como a aplicação e a infraestrutura são gerenciadas.
  - **Fase 5.2 - Checagem periódica**: checa a aplicação e a infraestrutura para que novos riscos sejam tratados e o nível de segurança seja preservado. É integrado com a gestão de riscos.
  - **Fase 5.3 - Verificação das mudanças**: checa a mudança para que o nível de segurança não seja afetado após a aprovação e teste da mudança em ambiente de teste e implantação em ambiente de produção. É integrado com a gestão de mudança.

Figura 3.22 | *Framework* da metodologia da OWASP



Fonte: adaptada de OWASP (2014).

A OWASP prevê algumas análises de segurança (OWASP, 2014):

- **Inspeção e revisão manual**: identifica preocupações de segurança com a análise de documentação e entrevistas, focando em como determinados pontos funcionam e como eles foram implementados. Esta técnica pode analisar o processo de ciclo de vida de desenvolvimento de *software* e assegurar que há uma política adequada e conhecida por todos, além das qualificações necessárias para a especificação e implementação da segurança na aplicação. É recomendado um modelo de "t*rust-but-verify*", já que nem tudo o que é mostrado ou dito é efetivo. As vantagens desta técnica são: não requer tecnologia de suporte; pode ser aplicada em situações variadas; tem flexibilidade; é aplicada no início do SDLC. As desvantagens são: pode consumir tempo; os materiais de suporte nem sempre estão disponíveis; requer competência e qualificação do profissional de segurança.

- **Modelagem de ameaças:** ajuda os arquitetos de sistemas a pensarem nas ameaças para seus sistemas e aplicações, possibilitando a criação de estratégias de mitigação para potenciais vulnerabilidades de uma forma priorizada. Os modelos de ameaças devem ser criados no início do SDLC e revisado conforme o progresso do desenvolvimento. As vantagens são: provê visão do sistema sob o ponto vista do atacante; tem flexibilidade; é aplicado no início do SDLC. As desvantagens são: técnica relativamente nova; modelos de ameaça bons não significam *softwares* bons. O modelo de ameaça pode ser desenvolvido de acordo com os passos do NIST 800-30 (NIST, 2012):
  - Decomposição da aplicação: entendimento de como a aplicação funciona, seus ativos, funcionalidades e conectividade.
  - Definição e classificação dos ativos: classificação dos ativos e priorização de acordo com a importância de negócio.
  - Exploração de vulnerabilidades potenciais: incluindo as técnicas, operacionais e de gerenciamento.
  - Exploração de ameaças potenciais: criação de visões de vetores de ataques potenciais com uso de cenários ou árvores de ataques.
  - Criação de estratégias de mitigação: controles de segurança para cada ameaça explorável.
- **Revisão de código-fonte:** é uma técnica capaz de detectar muitas vulnerabilidades que outras técnicas não permitem, já que todo problema de segurança está em algum ponto do código. Com a análise do código-fonte, o profissional pode determinar o que está ocorrendo com a aplicação e remover as possibilidades que aparecem em testes de caixa preta. As vantagens são: completude e efetividade; acurácia; rapidez na execução. As desvantagens são: requer qualificação de segurança dos desenvolvedores; pode não identificar problemas em bibliotecas compiladas; não detecta erros

de execução facilmente; necessita de análise de procedimentos operacionais, pois o código implantado pode não ser o mesmo do que está sendo analisado.

- **Teste de penetração ou *pentest*:** realizado de forma remota, do ponto de vista dos usuários, com o profissional atuando como um atacante, a partir de uma conta válida de usuário. Diferentemente de *pentests* em redes ou sistemas operacionais, que têm vulnerabilidades conhecidas e ferramentas automatizadas, os *pentests* em aplicações são mais complexos, já que possuem vulnerabilidades não conhecidas. A recomendação, assim, é que para as aplicações *web*, o *pentest* não seja utilizado como teste primário, apesar de ser útil para detectar algumas vulnerabilidades específicas que podem ser corrigidas. As vantagens são: pode ser rápido (e mais barato); requer menos qualificação do que uma revisão de código-fonte. As desvantagens são: é feito somente no final do SDLC; consegue testar somente a entrada.

> **REFLITA**
>
> Dentre os testes de segurança, o melhor é uma abordagem balanceada, que inclui diferentes técnicas, da revisão manual aos testes técnicos, em diferentes fases do SDLC. Em alguns casos, não há o acesso ao código-fonte, o que faz com que o *pentest* seja melhor do que nenhum teste, que pode ser ainda complementado com outros testes de segurança (OWASP, 2014).

METODOLOGIA OSSTMM

A metodologia *Open Source Security Testing Methodology Manual* (OSSTMM), da *Institute for Security and Open Methodologies* (ISECOM), surgiu em 2000 como um *framework* de melhores práticas e em 2005 evoluiu para uma metodologia. Em 2006, a OSSTMM se tornou um

padrão que foca na segurança, além de poder ser utilizado para a conformidade de acordo com um regulamento ou legislação específica (ISECOM, 2010).

A OSSTMM engloba cinco dimensões ou canais: humano, físico, sem fio, telecomunicações e redes de dados. Eles possibilitam testes de segurança na computação em nuvem, infraestruturas virtuais, *middleware* de mensagens, infraestruturas de comunicação móvel, locais de alta segurança, recursos humanos, computação confiável, e qualquer processo lógico que necessite de testes de segurança. A metodologia ainda tem um conjunto de métricas de superfície de ataque. Os testes podem ser certificados, de acordo com requisitos que incluem a condução dos testes, se todos os canais necessários foram testados, a postura dos testes de acordo com a lei, a mensuração dos resultados de uma forma quantificável, a consistência e repetitividade dos resultados, e se os resultados contêm apenas fatos derivados dos próprios testes (ISECOM, 2010).

Os sete passos para fazer um teste de segurança, segundo a OSSTMM (ISECOM, 2010), são:

- Definir o que será protegido, os ativos e os controles.
- Identificar a área em torno dos ativos que incluem mecanismos de proteção, os processos e serviços. Esta área é a zona de engajamento, onde ocorrem as interações com os ativos.
- Definir tudo o que está fora da zona de engajamento que é necessário para manter os ativos operacionais. Podem ser incluídos elementos que não podem ser influenciados diretamente, como eletricidade, alimento, água, ar, informação, legislação ou regulamentos. Podem ser incluídos ainda elementos que podem ser trabalhados, como a umidade, temperatura, claridade, fornecedores, parceiros, entre outros. E, também, podem ser incluídos processos, protocolos e recursos que mantêm a infraestrutura funcionando. Este é o escopo do teste.

- Definir como o escopo interage entre os elementos e fora dele. Vetores – que são compartimentos lógicos dos ativos que possuem as direções das interações, como de fora para dentro, de departamento A para departamento B, ou de dentro para dentro – devem ser utilizados.
- Identificar os equipamentos necessários para cada teste, em cada vetor, que por sua vez possui diferentes dimensões ou canais: humano, físico, sem fio, telecomunicações e redes de dados.
- Determinar quais informações serão geradas com os testes.
- Assegurar que o teste de segurança está em conformidade com as regras de engajamento, a fim de assegurar que o processo seja executado sem criar mal entendimentos, concepções equivocadas ou falsas expectativas.

Os tipos de testes de segurança previstos e citados no OSSTMM são seis (Figura 3.23), e são baseados na quantidade de informações que os profissionais têm dos alvos, o que o alvo sabe sobre o profissional ou a expectativa do teste e a legitimidade do teste. Alguns irão testar as qualificações do professional mais do que testar a segurança do alvo (ISECOM, 2010):

- ***Blind:*** o profissional testa o alvo sem conhecimento prévio sobre a defesa, os ativos e os canais. O alvo é preparado para a análise com o conhecimento dos detalhes da análise. Este teste visa principalmente a qualificação do profissional, já que os avanços da análise podem ir até onde essa qualificação permite. Também é conhecido como *ethical hacking*, ou *war gaming* ou *role playing* no contexto do canal físico.
- ***Double Blind:*** o profissional testa o alvo sem conhecimento prévio sobre a defesa, os ativos e os canais. O alvo não é notificado sobre o escopo da análise, nem dos canais a serem testados e nem dos vetores de testes. Este teste visa a qualificação do profissional e a preparação do alvo para variáveis desconhecidas. Os avanços da

análise podem ir até onde a qualificação do profissional permite. Também é conhecido como teste de caixa preta ou *pentest*.

- ***Gray Box:*** o profissional testa o alvo com conhecimento limitado sobre a defesa e os ativos e conhecimento total dos canais. O alvo é preparado para a análise com o conhecimento dos detalhes da análise. Este teste visa a qualificação do profissional. A natureza do teste é a eficiência. Os avanços da análise podem ir até onde a qualificação do profissional permite e dependem da qualidade das informações providas para o profissional antes da análise. Também é conhecido como teste de vulnerabilidade e é iniciado pelo alvo como uma autoavaliação.

- ***Double Gray Box:*** o profissional visa o alvo com um conhecimento limitado sobre a defesa e os ativos e conhecimento total dos canais. O alvo é notificado sobre o escopo e o período da análise, mas não sobre os canais e vetores de teste. Este teste visa a qualificação do profissional e a preparação do alvo. Os avanços da análise podem ir até onde a qualificação do profissional permite e dependem da qualidade das informações providas para o profissional e para o alvo antes da análise. Também é conhecido como teste de caixa branca.

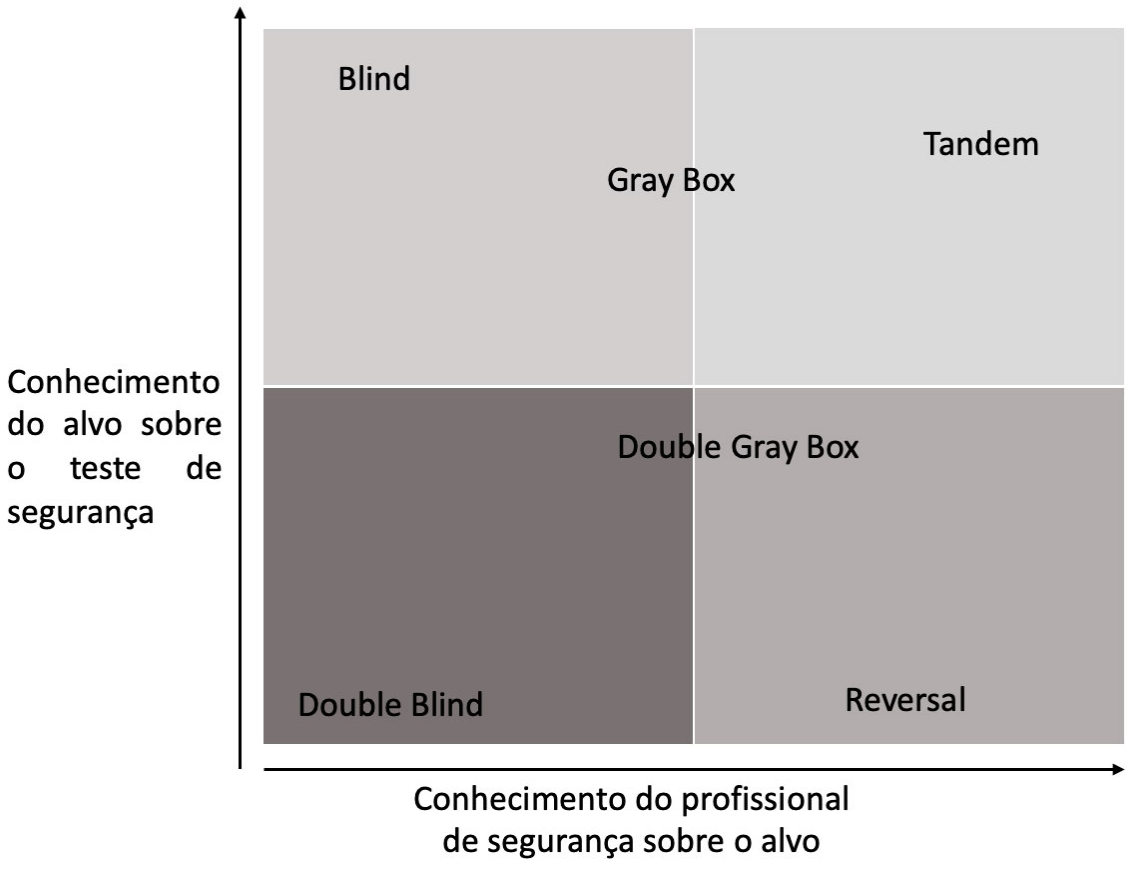
- ***Tandem:*** o profissional e o alvo são preparados para a análise, tendo conhecimento sobre os detalhes do teste. Este teste visa a proteção e os controles do alvo. No entanto, não testa o estado de preparação do alvo. A natureza do teste é a meticulosidade, já que o analista não tem a visão completa dos testes e suas respostas. Os avanços da análise podem ir até onde a qualificação do profissional permite, e dependem da qualidade das informações providas para o profissional antes da análise. Também é conhecido como *in-house audit* ou teste de *crystal box*, e o profissional é geralmente parte do processo de segurança.

- ***Reversal:*** o profissional visa o alvo com conhecimento total sobre os processos e segurança operacional, mas o alvo não sabe sobre o

que, como, e quando o profissional irá fazer o teste. A natureza do teste é a análise do estado de preparação do alvo. Os avanços da análise podem ir até onde a qualificação e criatividade do profissional permitem e dependem da qualidade das informações providas para o profissional antes da análise. Também é conhecido como exercício de *red team.*

Ver anotações 0

Figura 3.23 | Tipos de testes do OSSTMM

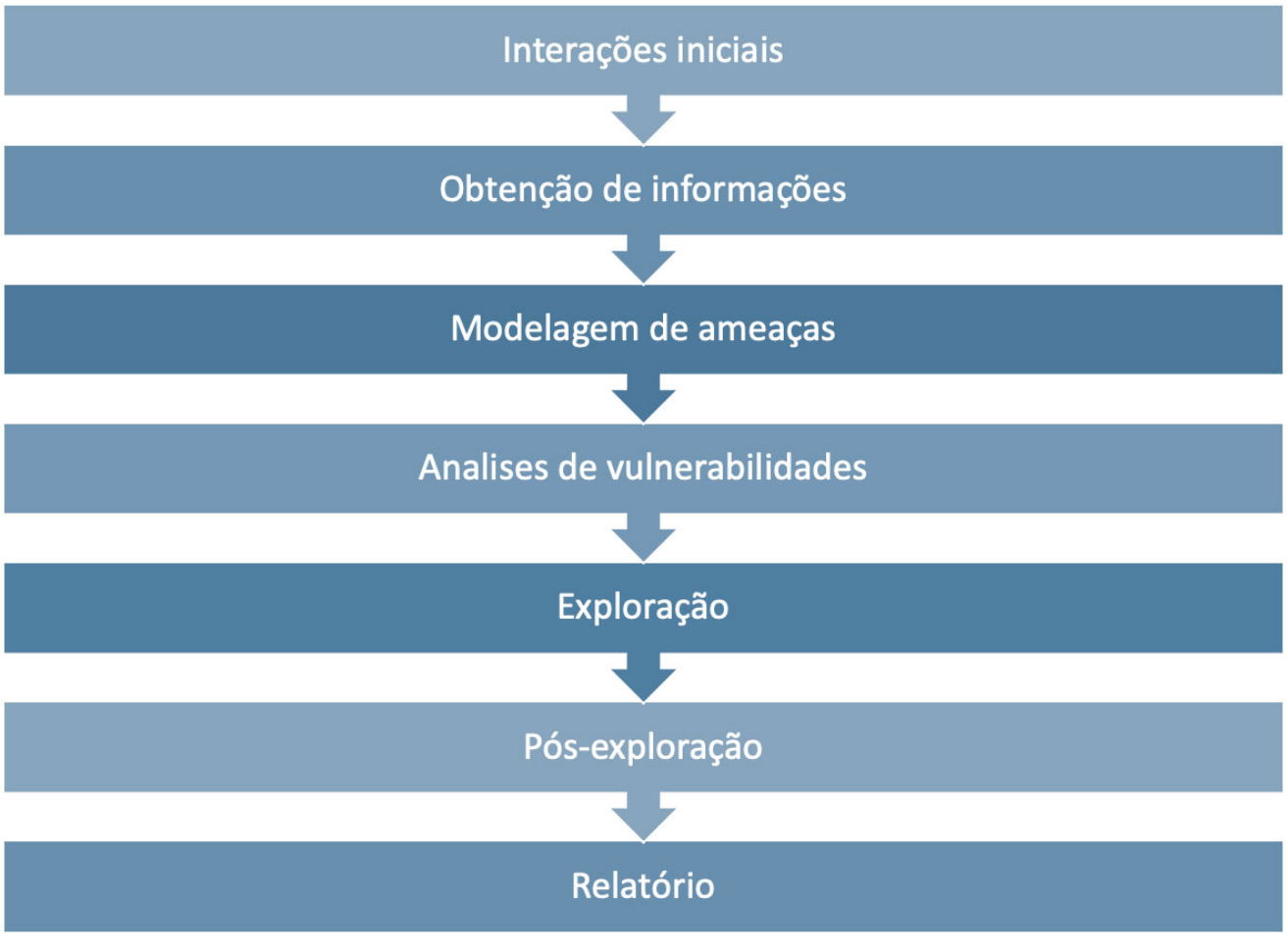


Fonte: adaptado de ISECOM (2010).

METODOLOGIA PTES

A metodologia *Penetration Testing Execution Standard* (PTES) é composta por sete seções (Figura 3.24), que definem as atividades a serem realizadas, desde as interações iniciais até o relatório. De uma forma geral, as atividades são suportadas por uma documentação técnica detalhada, para cada uma das seções do PTES. As seções descrevem como iniciar as atividades, obter informações para a análise, a modelagem de ameaças, as análises de vulnerabilidades, a exploração para passar pelos controles de segurança existentes, o pós-exploração para manter o acesso e controle do alvo e o relatório final.

Figura 3.24 | Metodologia PTES



Fonte: adaptada de PTES (2014).

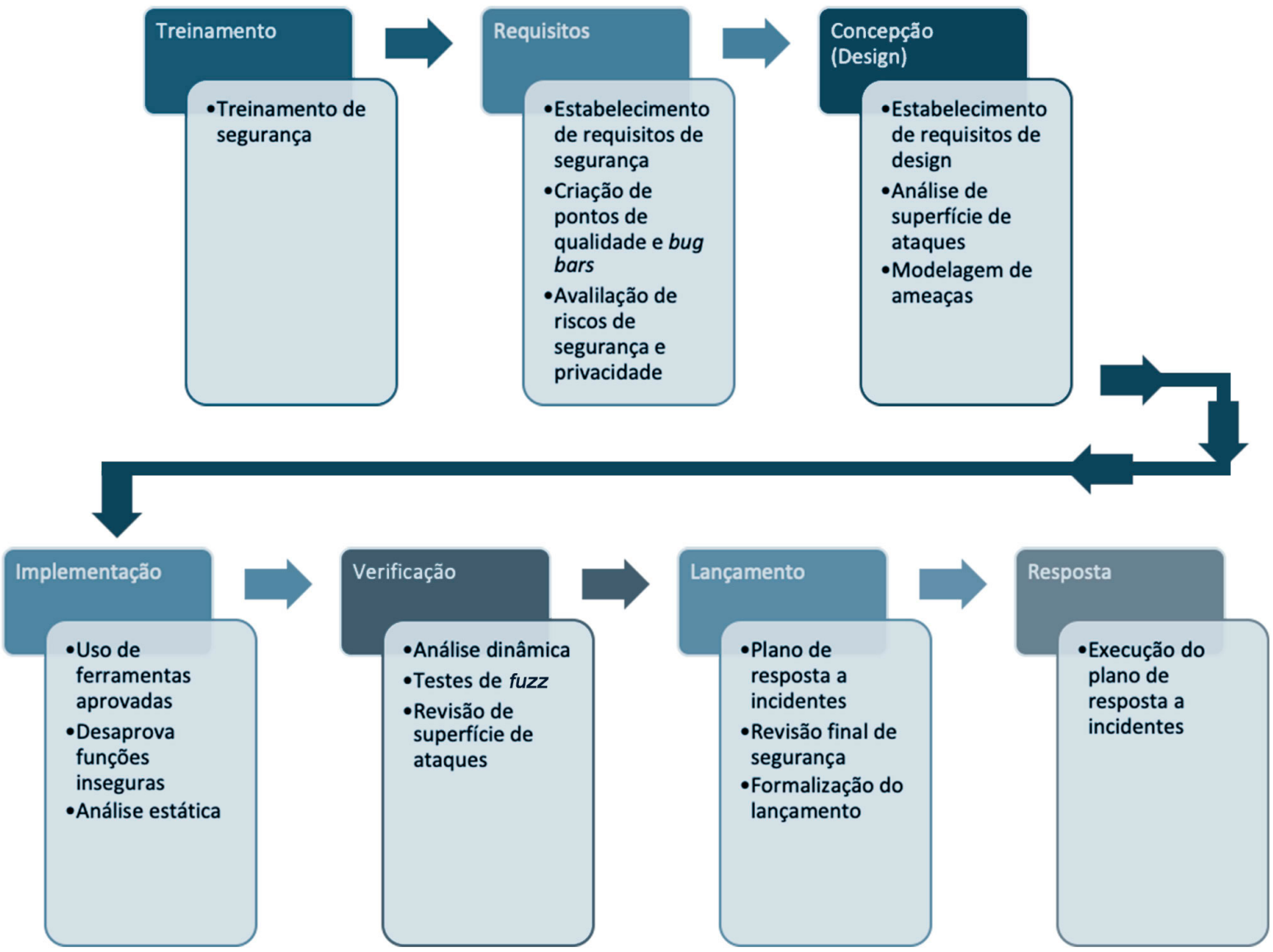
APLICAÇÃO DOS TESTES DE SEGURANÇA

Os testes de segurança são parte importante da gestão de segurança da informação e devem ser feitos por todas as empresas, sejam aquelas que desenvolvem *software* ou aquelas que adquirem sistemas.

A metodologia OWASP *Testing Project* foca na segurança no ciclo de vida de desenvolvimento de *software*, com a inserção de testes de segurança em diferentes pontos do ciclo. O desenvolvimento seguro envolve ainda outros elementos de segurança como o treinamento de segurança, o estabelecimento de requisitos de segurança, a criação de pontos de qualidade, a inclusão de funções de segurança, a avaliação de riscos de segurança e privacidade e o plano de resposta a incidentes após a implantação (Figura 3.25).

Figura 3.25 | Ciclo de vida de desenvolvimento seguro

Ver anotações



Fonte: adaptada de Lipner (2010).

SAIBA MAIS

As análises de código-fonte, realizadas no *Static Analysis Security Testing* (SAST) podem ser feitas com o uso de ferramentas ou manualmente. As ferramentas conseguem identificar erros no código, mas dificilmente conseguem identificar falhas na especificação e na lógica.

E o teste de caixa preta pode também ter algumas limitações na identificação de vulnerabilidades. Um exemplo é o uso de um mecanismo criptográfico para autenticar um usuário de diferentes *sites*. Neste exemplo, um usuário autenticado no *site* A pode visitar o *site* B automaticamente. Nesta implementação, essa validação é feita com o uso de um *hash* do nome de usuário e data, que é enviado ao site B pelo site A e pelo usuário. O site B pode então comparar o *hash* para validar o usuário. O problema de segurança é que, uma vez descoberto o funcionamento, qualquer agente de ameaça que captura o *hash* pode chegar ao site B. O teste de caixa

Ver anotações

preta enxerga o *hash*, sem saber a sua função, de uma forma direta, que só pode ser identificada com uma análise de código.

PESQUISE MAIS

Aplicações *web* têm características diferentes de aplicações móvel. Os testes de segurança seguem as principais fases de uma forma genérica, mas os testes específicos são diferentes, dependendo do ambiente. E é aí que entra a qualificação do profissional de segurança, que deve fazer os testes condizentes com o ambiente que está sendo analisado.

Lembre-se que a qualidade dos testes vai até onde chega a capacidade do profissional, incluindo o limitante do tempo disponível.

A OWASP Testing Project (2014) apresenta exemplos práticos e as atividades de como executar os testes, além da indicação de ferramentas e referências sobre cada teste. Dentre as categorias de testes, estão:

- Obtenção de informação.
- Gerenciamento de configuração e implantação.
- Gestão de identidades.
- Autenticação.
- Autorização.
- Gerenciamento de sessão.
- Validação de entradas.
- Criptografia.
- Lógica de negócio.
- Lado cliente.

OWASP. **Testing Guide 4.0.**

Assim chegamos ao final desta seção que trata de testes de segurança. Você já conhece os diferentes tipos de testes que podem ser realizados, tanto para sistemas adquiridos quanto para *software* que é desenvolvido. Menos vulnerabilidades significam menos chances de exploração dos ativos de sua empresa, ou seja, menos incidentes de segurança.

Ver anotações

FAÇA VALER A PENA

Questão 1

Testes de segurança são importantes para identificar as vulnerabilidades de um sistema, para que assim possam ser tratadas, antes de serem exploradas por agentes de ameaça. Quando a exploração ocorre, os resultados são incidentes de segurança e impactos para a organização.

O tipo de teste de segurança que é realizado sob o ponto de vista de um usuário ou agente de ameaça, em que nenhuma informação prévia do ambiente é fornecida para o profissional de segurança é o:

a. Teste de caixa preta.

b. Teste de caixa cinza.

c. Teste de caixa branca.

d. *Static Analysis Security Testing* (SAST).

e. *Dynamic Analysis Security Testing* (DAST).

Questão 2

Sua empresa está finalizando o desenvolvimento de um sistema e você faz parte da equipe de desenvolvimento, focando em aspectos de segurança e privacidade. Você já fez uma análise de código-fonte e agora precisa testar o sistema com ele funcionando.

O teste de segurança que é feito neste estágio do ciclo de vida de desenvolvimento de software, antes da implantação é o:

a. Teste de caixa preta.

b. Teste de caixa cinza.

c. Teste de caixa branca.

d. *Static Analysis Security Testing* (SAST).

e. *Dynamic Analysis Security Testing* (DAST).

Questão 3

Testes de segurança podem ser feitos a partir do ambiente interno ou a partir do ambiente externo. Há testes que analisam o código-fonte e outros que analisam o ambiente em execução. O objetivo é sempre utilizar e desenvolver sistemas com o mínimo de vulnerabilidades, já que são elas que são exploradas em ataques.

Assinale a alternativa que contém os testes de segurança que você pode realizar e que fazem uso de código-fonte.

a. Pentest de caixa preta e *Dynamic Analysis Security Testing* (DAST).

b. *Static Analysis Security Testing* (SAST) e *Dynamic Analysis Security Testing* (DAST).

c. Pentest de caixa preta e *Static Analysis Security Testing* (SAST).

d. Pentest de caixa cinza e *Static Analysis Security Testing* (SAST).

e. Pentest de caixa branca e *Static Analysis Security Testing* (SAST).

REFERÊNCIAS

CAVALANCIA, N. Vulnerability management explained, Security Essentials, **AT&T Cybersecurity**, 2 jul. 2020. Disponível em: http://soc.att.com/3tOlEZi. Acesso em: 31 dez. 2020.

CONSTANTIN, L. O que é o DevSecOps? Por que é difícil fazer? **SegInfo**, 31 jul. 2020. Disponível em: https://bit.ly/31sTl7T. Acesso em: 20 jan. 2021.

GOMES, P. C. T. O que é um hackathon e como pode beneficiar a sua empresa? Inovação e Tecnologia, **OPServices**, 12 jan. 2017. Disponível em: https://bit.ly/2NNX3We. Acesso em: 20 jan. 2021.

Ver anotações 0

HICKEN, A. Software Safety and Security Through Standards, **Parasoft**, 15 set. 2016. Disponível em: https://bit.ly/3d22euq. Acesso em: 2 nov. 2020.

ISECOM, Institute for Security and Open Methodologies. **OSSTMM 3** – The Open Source Security Testing Methodology Manual. 2010. Disponível em: https://bit.ly/39b6KWn. Acesso em: 27 dez. 2020.

KOUSSA, S. What Do Sast, Dast, Iast And Rasp Mean To Developers? **SoftwareSecured**, 2 nov. 2018. Disponível em: https://bit.ly/31dI2QP.Acesso em: 9 dez. 2020.

LIPNER, S. Microsoft Corporation. The OWASP Foundation. **The Security Development Lifecycle**. 24 jun. 2010. Disponível em: https://bit.ly/2QAHeDs. Acesso em: 2 nov. 2020.

MENA, I. Verbete Draft: o que é Capture The Flag (CTF). **Draft**, 7 fev. 2018. Disponível em: https://bit.ly/3vRUrrL. Acesso em: 20 jan. 2021.

MICROSOFT Corporation. **What are the Microsoft SDL practices?** Disponível em: https://bit.ly/39fnJqy. Acesso em: 2 nov. 2020.

NAKAMURA, E. T. **Segurança da informação e de redes.** Editora e Distribuidora Educacional S.A., 2016.

NIST, National Institute of Standards and Technology, U.S. Department of Commerce. Joint Task Force Transformation Initiative. **NIST Special Publication 800-30 Revision 1**. Guide for Conducting Risk Assessments. 12 set. 2020. Disponível em: https://bit.ly/3lOkKKG. Acesso em: 31 dez. 2020.

OWASP. **Intermediate update 1.1.3** (OSS Release), 4 ago. 2019. Disponível em: https://bit.ly/3fciKuk. Acesso em: 26 dez. 2020.

OWASP. **Testing Guide 4.0**. Disponível em: https://bit.ly/3d5Or5U. Acesso em: 28 dez. 2020.

PCI Data Security Standard (PCI DSS) 1.1. Penetration Test Guidance Special Interest Group PCI Security Standards Council. **Information Supplement**: Penetration Testing Guidance. Set. 2017. Disponível em: https://bit.ly/3snNtbF. Acesso em: 30 dez. 2020.

PTES. **Main page**. 16 ago. 2014. Disponível em: https://bit.ly/3faCkao. Acesso em: 30 dez. 2020.

TSE. **TSE realiza teste público de segurança do sistema eletrônico de votação de 28 a 30 de novembro**. 27 nov. 2017. Disponível em: https://bit.ly/39dWRqJ. Acesso em: 20 jan. 2021.

WARBURTON, A. Bug Bounty: como funciona o mundo do *hacking* ético e a caça às vulnerabilidades. **Welivesecurity**, 23 jan. 2020. Disponível em: https://bit.ly/2PqDtQc. Acesso em: 20 jan. 2021.