International Software Testing Qualifications Board

ISTQB® Advanced Level

Security Tester

Versão 2016br

Comissão Internacional para Qualificação de Teste de *Software*

Tradução realizada pela WG Traduções do BSTQB baseada na versão 2016 do *Advanced Level Certified Security Tester*do ISTQB.

Brazilian Software Testing Qualifications Board

Este documento pode ser copiado na sua totalidade, ou ter extratos feitos, se a fonte for reconhecida na sua reprodução.
Copyright © International Software Testing Qualifications Board (ISTQB®).
Advanced Level WG: Mike Smith (chair)
Advanced Security Tester Syllabus WG: Randall Rice (chair), Tarun Banga, Taz Daughtrey, Frans Dijkman, Prof. Dr. Stefan Karsch, Satoshi Masuda, Raine Moilanen, Joel Oliveira, Alain Ribault, Ian Ross, Kwangik Seo, Dave van Stein, Dr. Nor Adnan Yahaya, Wenqiang Zheng.

Histórico de Revisões

Versão	Data	Descrição
0.1	24/04/2015	Baseline version created from existing Expert Security Tester draft syllabus version 3.9
0.2	15/06/2015	Consolidated author input after Oslo author meeting
1.0 - Beta	20/09/2015	Beta release – alpha release comments incorporated
1.0 – GA Candidate	04/03/2016	After Exam WG review, changed LO 4.1.2 from K2 and K3 and re-worded appropriately. Text already adequately supports a K3 LO.
1.0 - GA	18/03/2016	GA release – beta release comments incorporated
2016br	15/03/2017	Tradução para a Língua Portuguesa
2016br r1	09/02/2018	Revisão geral

Índice

Agradecimentos

Este documento foi produzido pela equipe do Grupo de Trabalho de Nível Avançado do *International Software Testing Qualifications Board*.

A equipe principal agradece à equipe de revisão e a todos os Conselhos Nacionais por suas sugestões.

No momento em que o Syllabus de Nível Avançado para este módulo foi concluído, o Grupo de Trabalho de Nível Avançado - Security Tester teve a seguinte associação:

Os autores da equipe principal para este plano de estudos: Randall Rice (presidente), Hugh Tazwell Daughtrey (vice-presidente), Frans Dijkman, Joel Oliveira, Alain Ribault.

As seguintes pessoas participaram da revisão, comentários e votação deste programa (Ordem alfabética): Tarun Banga, Clive Bates, Hugh Tazwell Daughtrey (vice-presidente), Frans Dijkman (autor), Christian Alexander Graf, Wenda Hu, Matthias Hamburgo, Prof. Dr. Stefan Karsch, Sebastian Malyska, Satoshi Masuda, Gary Mogyorodi, Raine Moilanen, Joel Oliveira, Meile Posthuma, Alain Ribault, Randall Rice (presidente), Ian Ross, Kwangik Seo, Dave van Stein, Ernst von During, Attila Toth, Wei Xue, Dr. Nor Adnan Yahaya, Xiaofeng Yang, Wenqiang Zheng, Ping Zuo.

Além disso, reconhecemos e agradecemos aos líderes e membros do Grupo de Trabalho de Nível Expert pela sua contínua orientação: Graham Bath (Presidente, Grupo de Trabalho de Nível Expert), Judy McKay (Vice-Presidente, Grupo de Trabalho de Nível Expert).

Este documento foi formalmente divulgado pela Assembleia Geral do ISTQB® em 18 de março de 2016.

0 Introdução a este Syllabus

0.1 Objetivos deste documento

Este programa forma a base para a Certificação do *ISTQB® Advanced Level - Security Tester*. O ISTQB® fornece este programa da seguinte forma:

1. Aos Conselhos Nacionais, traduzir para o seu idioma local e credenciar os fornecedores de treinamento. Os conselhos nacionais podem adaptar o programa às suas necessidades linguísticas específicas e modificar as referências para se adaptarem às suas publicações locais.
2. Para os Conselhos de Exame, para derivar questões de exame em sua língua local com base nos objetivos de aprendizagem para cada módulo.
3. Para os provedores de treinamento, produzir material didático e determinar métodos de ensino apropriados.
4. Para os candidatos de certificação, como uma fonte para se preparar para o exame.
5. Para a comunidade internacional de software e engenharia de sistemas, para avançar a profissão de software e testes de sistema, e como base para livros e artigos.

O ISTQB® pode permitir que outras entidades usem este programa para outros fins, desde que obtenham autorização prévia por escrito.

0.2 Visão geral

A certificação *Advanced Level Security Tester* destina-se a pessoas que já alcançaram um ponto avançado em suas carreiras em testes de software e desejam desenvolver ainda mais sua experiência em testes de segurança. Os módulos oferecidos no Nível Avançado cobrem uma ampla gama de tópicos de teste.

Para receber a Certificação do *ISTQB® Advanced Level - Security Tester*, os candidatos devem possuir o certificado CTFL e satisfazer a Qualificação junto ao BSTQB para realizar exames CTAL. Consulte o site do BSTQB para conhecer os critérios específicos de experiência prática para certificações CTAL.

0.3 Exames

Todos os exames realizados no Nível Avançado para este módulo são baseados neste Syllabus. O formato do exame é definido pelas Diretrizes de Exames Avançados do ISTQB.

Os exames podem ser tomados como parte de um curso de treinamento acreditado ou realizados independentemente (por exemplo, em um centro de exames). Exames podem ser feitos em papel ou eletronicamente, mas todos os exames devem ser supervisionados / observados (supervisionados por uma pessoa mandatada por um Conselho Nacional ou de Exame).

0.4 Como este syllabus é organizado

Há dez capítulos. O título de nível superior mostra a quantidade de minutos que devem ser dedicados a estudo de cada capítulo. Por exemplo:

1. A Base de Testes de Segurança (105 min)

Mostra que o Capítulo 1 pretende ter um tempo de 105 minutos de dedicação ao estudo deste capítulo.Os objetivos de aprendizagem específicos são listados no início de cada capítulo.

0.5 Definições

Muitos termos usados na literatura de software são comuns à todas as certificações. Considerando que os candidatos dos níveis Fundamental e Avançado podem responder questões baseadas apenas no Glossário de Termos, é esperado, além disso, que os candidatos deste nível devem estar cientes e capazes de trabalhar com as diferentes definições.

NOTA: A "garantia de informação" (IA) é chamada apenas na seção 2.4. Uma citação em 2.4.3 é seguida pela asserção de que IA deve ser visto como mais amplo do que "testes de segurança" da mesma forma que o QA é mais amplo do que o teste de software.

A "segurança da informação" é utilizada nas secções 2.2, 2.3.1, 2.7.2, 6 (Antecedentes), 6.1.3 e no Capítulo 9.

Não há uso do termo "segurança cibernética", que em alguns setores é agora referido como IA.

As palavras-chave listadas no início de cada capítulo neste programa de Nível Avançado são definidas no Glossário Padrão de Termos usados em Testes de Software, publicado pelo ISTQB, ou são fornecidos na literatura referenciada.

0.6 Nível de detalhe

O nível de detalhe neste programa permite um aprendizado e um exame consistentes internacionalmente. Para atingir esse objetivo, o programa de estudos consiste em:

- Objetivos gerais de instrução descrevendo a intenção do Nível Avançado
- Objetivos de aprendizagem para cada área de conhecimento, descrevendo o resultado de aprendizagem cognitiva e mentalidade a ser alcançada
- Uma lista de informações a aprender, incluindo uma descrição e referências a fontes adicionais, se necessário
- Uma descrição dos conceitos-chave para aprender, incluindo fontes como literatura ou padrões aceitos
- Algumas ferramentas, métodos e marcas podem ser mencionados neste programa. Este programa não se destina a promover ou recomendar qualquer solução de segurança específica.

O conteúdo do programa não é uma descrição de toda a área de conhecimento para Testadores em segurança Avançados, ele reflete o nível de detalhe a ser coberto em um curso *Advanced Security Tester*.

0.7 Objetivos de aprendizado e níveis de conhecimento

O conteúdo deste programa, os termos e os principais elementos (fins) de todas as normas listadas devem pelo menos ser lembrados (K1) e compreendidos (K2), mesmo que não mencionados explicitamente nos objetivos de aprendizagem.

Os seguintes objetivos de aprendizagem são definidos como aplicáveis a este programa. Cada tópico no programa será examinado de acordo com o objetivo de aprendizagem para ele.

0.7.1 Nível 1: Lembre-se (K1)

O candidato reconhecerá, lembrará e recordará um termo ou conceito.

Palavras-chave: Lembrar, recordar, reconhecer, saber. Exemplo

Pode reconhecer a definição de "risco" como: "*um fator que poderia resultar em consequências negativas futuras; geralmente expressa como impacto e probabilidade*".

0.7.2 Nível 2: Compreender (K2)

O candidato pode selecionar as razões ou explicações para declarações relacionadas ao tópico e pode resumir, diferenciar, classificar e dar exemplos de fatos (por exemplo, comparar termos), os conceitos de teste, procedimentos de teste (explicando a sequência de tarefas).

Palavras-chave: Resumir, classificar, comparar, mapear, contrastar, exemplificar, interpretar, traduzir, representar, inferir, concluir, categorizar.

Exemplos:

Explique a razão pela qual os testes de segurança devem ser concebidos o mais cedo possível:

- Encontrar defeitos de segurança e vulnerabilidades quando eles são mais baratos para evitar a construção de um sistema ou aplicativo propenso a patches contínuos de vulnerabilidades de segurança.

0.7.3 Nível 3: Aplicar (K3)

O candidato pode selecionar a aplicação correta de um conceito ou técnica e aplicá-lo a um dado contexto. K3 é normalmente aplicável ao conhecimento processual. Não há nenhum ato criativo envolvido, como avaliar um aplicativo de software ou criar um modelo para um determinado programa de software. Quando um modelo é fornecido, o programa explica as etapas processuais necessárias para criar casos de teste a partir desse modelo, então é K3.

Palavras-chave: Implementar, executar, usar, seguir um procedimento, aplicar um procedimento.

Exemplos:

- Utilize o procedimento genérico para a criação de casos de teste de segurança para selecionar os casos de teste de um dado diagrama de transição de estado para cobrir todas as transições.

0.7.4 Nível 4: Analisar (K4)

O candidato pode separar informações relacionadas a um procedimento ou técnica em suas partes constituintes para melhor compreensão, e pode distinguir entre fatos e inferências. Aplicação típica é analisar um documento, software, situação do projeto e propor ações adequadas para resolver um problema ou tarefa.

Palavras-chave: analisar, diferenciar, selecionar, estrutura, foco, atributo, desconstruir, avaliar, julgar, monitorar, coordenar, criar, sintetizar, gerar, hipótese, plano, projeto, construir, produzir.

Exemplos:

- Analisar os riscos de segurança do produto e propor atividades preventivas e corretivas de mitigação.
- Selecione as ferramentas de teste de segurança que seriam mais apropriadas em uma determinada situação com falhas de segurança passadas.

Referência (Para os níveis cognitivos dos objetivos de aprendizagem)

- Bloom, B. S. (1956). Taxonomia de Objetivos Educacionais, Manual I: O Domínio Cognitivo, David McKay, Co. Inc.
- Anderson, L. W. e Krathwohl, D. R. (eds) (2001). Uma Taxonomia para Aprender, Ensinar e Avaliar: Uma Revisão da Taxonomia de Bloom de Objetivos Educacionais, Allyn & Bacon.

1 A base do teste de segurança (105 min)

Palavras-chave

Privacidade do dado, hacker ético, segurança da informação, teste de penetração, avaliação de risco, exposição ao risco, mitigação do risco, ataque de segurança, auditoria de segurança, política de segurança, procedimento de segurança, risco de segurança.

Objetivos de aprendizagem

1.1 Riscos de segurança

AS-1.1.1 (K2): Compreender o papel da avaliação de risco no fornecimento de informações para o planejamento de testes de segurança e modelagem, e alinhamento de testes de segurança com as necessidades do negócio.

AS-1.1.2 (K4): Identificar os ativos significativos a serem protegidos, o valor de cada ativo e os dados necessários para avaliar o nível de segurança necessário para cada ativo.

AS-1.1.3 (K4): Analisar a utilização eficaz das técnicas de avaliação de riscos numa dada situação para identificar as ameaças atuais e futuras à segurança.

1.2 Políticas e procedimentos de segurança da informação

AS-1.2.1 (K2): Compreender o conceito de políticas e procedimentos de segurança e como eles são aplicados em sistemas de informação.

AS-1.2.2 (K4): Analisar um determinado conjunto de políticas e procedimentos de segurança, juntamente com os resultados dos testes de segurança para determinar a eficácia.

1.3 A auditoria de segurança e seu papel no teste de segurança

AS-1.3.1 (K2): Compreender a finalidade de uma auditoria de segurança.

Os testes funcionais baseiam-se em uma variedade de itens, como riscos, requisitos, casos de uso e modelos. Os testes de segurança baseiam-se nos aspectos de segurança dessas especificações, mas também buscam verificar e validar os riscos de segurança, procedimentos e políticas de segurança, comportamento do invasor e vulnerabilidades de segurança conhecidas.

1.1 Riscos de segurança

1.1.1 O papel da avaliação de risco em testes de segurança

Os objetivos do teste de segurança são baseados em riscos de segurança. Estes riscos são identificados através da realização de uma avaliação de risco de segurança. Técnicas gerais de gestão de risco são descritas em [BSTQB_FL_SYL] e [BSTQB_ATM_SYL].

O risco é uma medida da extensão em que uma entidade é ameaçada por uma circunstância ou evento em potencial, e é tipicamente uma função de:

- Impactos adversos que ocorreriam se a circunstância ou evento ocorrer,
- Probabilidade da ocorrência.

Os riscos de segurança da informação são aqueles que resultam da perda de confidencialidade, integridade ou disponibilidade de informações ou sistemas de informação e refletem os impactos adversos potenciais às operações organizacionais (i.e., missão, funções, imagem ou reputação), ativos organizacionais, outras organizações e um país. [NIST 800-30]

O papel de uma avaliação de risco de segurança é permitir que uma organização compreenda quais áreas e ativos podem estar em risco e determinar a magnitude de cada risco. Para os testadores em segurança, uma avaliação de risco de segurança pode ser uma rica fonte de informações a partir da qual os testes de segurança podem ser planejados e projetados. Além disso, uma avaliação de risco de segurança pode ser usada para priorizar testes de segurança para que o mais alto nível de rigor de teste e cobertura, pode ser focada nas áreas com a grande exposição ao risco.

Ao priorizar os testes de segurança com base em uma avaliação de risco de segurança, os testes ficam alinhados com os objetivos de segurança do negócio. No entanto, para que esse alinhamento ocorra, a avaliação de risco de segurança deve refletir com precisão as ameaças de segurança da organização, as partes interessadas impactadas e os ativos a serem protegidos.

É importante entender que qualquer avaliação de risco (de segurança ou de outra forma) é apenas um instantâneo em um dado momento e com base em informações limitadas que podem levar a suposições inválidas e conclusões. Riscos de segurança mudam continuamente dentro de uma organização e projetos desde novas ameaças surgem diariamente. Por conseguinte, as avaliações dos riscos de segurança devem ser efetuadas a intervalos regulares. O intervalo de tempo exato para a realização de avaliações de risco de segurança varia com base na organização e no grau de mudança que ela experimenta. Algumas organizações realizam avaliações de risco de segurança em uma base de três a seis meses, enquanto outros realizá-los em uma base anual.

Outra questão com as avaliações de risco é o nível de conhecimento dos participantes. Os riscos podem ser perdidos devido à falta de informações detalhadas. Além disso, os riscos podem ser perdidos se as pessoas não entenderem ameaças e riscos de segurança. Por esta razão, é bom para solicitar a entrada de uma variedade de pessoas e prestar muita atenção ao nível de detalhe contido na informação que eles fornecem.

É uma expectativa realista que as suposições erradas podem ser feitas que podem conduzir aos riscos de segurança importantes que são faltados na avaliação. Maneiras de lidar com a possibilidade de falta ou incompleta de informações de risco incluem o uso de uma metodologia de avaliação de risco de segurança estabelecida como uma lista de verificação e recebendo a entrada de várias pessoas. Uma tal metodologia pode ser encontrada em [NIST 800-30].

1.1.2 Identificação de ativos

Nem todas as informações a serem protegidas estão em formato digital, como documentos físicos (contratos, planos, notas escritas, *logins* e senhas em forma escrita). Embora não estando em formato digital, esta informação pode ter alto valor. Portanto, a pergunta precisa ser feita, '*qual informação é digital e qual não é?*'. Talvez o bem a ser protegido exista tanto em formato digital quanto físico. Ao identificar os ativos a serem garantidos, devem ser feitas as seguintes perguntas:

Quais ativos são valiosos para a organização?

Exemplos de informações sigilosas de alto valor incluem:

- Dados do cliente.
- Planos de negócios.
- Software proprietário desenvolvido pela empresa.
- Documentação do sistema.
- Imagens e diagramas que são propriedade da empresa.
- Propriedade intelectual (por exemplo, processos, segredos comerciais).
- Planilhas financeiras.
- Apresentações e cursos de formação.
- Documentos.
- E-mails.
- Registros de funcionários.
- Declarações fiscais.

Embora muitos ativos sejam baseados na informação, é possível que alguns ativos em uma organização sejam de natureza física ou intangível. Exemplos desses ativos incluem:

- Protótipos físicos de novos dispositivos em desenvolvimento.
- A capacidade de prestar serviços.
- Reputação e confiança da empresa.

Qual o valor do ativo?

Muitos ativos sigilosos têm um valor tangível. Outros são medidos mais nos custos e nas consequências de sua perda. Por exemplo, o que um competidor faria com o plano de negócios de um concorrente?

Valor pode ser difícil de avaliar com certeza, entretanto, alguns métodos para determinar o valor de ativos digitais incluem:

- A receita futura a ser gerada pelo ativo.
- O valor para um concorrente que pode obter a informação.
- O tempo e o esforço necessário para recriar o ativo.
- Multas e penalidades por não poderem produzir a informação quando necessário, por exemplo, para uma auditoria ou ação judicial.
- Multas por perda de dados de clientes.

Onde estão localizados os ativos digitais?

No passado, os ativos digitais residiam em servidores, computadores de mesa ou periféricos, como discos ou CDs. Embora esta seja uma abordagem desatualizada e desorganizada, ainda pode haver dados sigilosos em CDs, DVDs e drives USB antigos. Um meio mais seguro de armazenar ativos digitais é o uso de servidores corporativos seguros, usando criptografia forte para todos os dados confidenciais. Para acessar dados confidenciais armazenados em servidores seguros, a autenticação e a autorização devem ser necessárias.

Além disso, pode ser necessária outra proteção de segurança, como certificados digitais para acessar informações confidenciais pela Internet.

A forma de armazenamento está sempre mudando. Agora, grandes quantidades de dados de negócios podem existir em dispositivos móveis, como *smartphones* e *tablets*. Se as informações digitais foram migradas para armazenamento em nuvem, haverá um novo conjunto de preocupações de segurança com base no acesso aos dados.

A importância da questão do armazenamento de dados surge de casos passados em que as pessoas de confiança, com acesso à dados sigilosos, simplesmente saíram de um edifício da empresa com um disco rígido cheio de informações sigilosas de clientes privados e de negócios. Um desses casos nos Estados Unidos envolveu um disco rígido roubado de uma área protegida em uma agência de segurança do governo, que incluía folha de pagamento e informações bancárias para mais de 100.000 trabalhadores atuais e antigos. [Washington Post, 2007].

Como são acessados os ativos digitais?

Os métodos comuns para acessar ativos digitais incluem:

- Acesso do computador através de uma rede de área local ou redes Wi-Fi.
- Acesso remoto através de uma rede privada virtual (VPN) ou unidade na nuvem (*cloud*).
- Passando armazenamentos físicos de dados (CDs, DVDs, *pendrivers*) de pessoa para pessoa, o que é uma prática muito comum.
- Envio de arquivos por e-mail.

Como os ativos digitais são protegidos?

Existem várias maneiras de proteger ativos digitais, incluindo:

- Criptografia (De qual tipo? Qual o tamanho da chave?).
- Autenticação e *tokens* (São necessários certificados digitais? As políticas de senha são adequadas e seguidas?).
- Autorização (Que níveis de privilégio foram concedidos aos usuários que lidam com ativos digitais?).

1.1.3 Análise das Técnicas de Avaliação de Risco

O processo de avaliação de riscos de segurança é muito semelhante a uma avaliação de risco padrão, com a principal diferença sendo o foco em áreas relacionadas à segurança.

Uma avaliação de risco de segurança deve incluir as perspectivas de partes interessadas no teste de segurança externa (ou seja, pessoas ou partes envolvidas no projeto/produto que estão fora da empresa e têm uma participação clara na segurança do projeto/produto). Estas partes interessadas incluem:

- Clientes e usuários para entender a perspectiva, obter entrada para testes de segurança e estabelecer boas comunicações.
- O público e a sociedade. Importante para transmitir que a segurança da informação é um esforço e uma responsabilidade da comunidade.
- Agências reguladoras. Necessárias para assegurar o cumprimento das leis aplicáveis em matéria de segurança da informação.

A preparação para uma avaliação de risco inclui as seguintes tarefas [NIST 800-30]:

- Identificar o objetivo da avaliação.
- Identificar o escopo da avaliação.
- Identificar os pressupostos e constrangimentos associados à avaliação.
- Identificar as fontes de informação a serem utilizadas como insumos para a avaliação.

- Identificar o modelo de risco e abordagens analíticas (isto é, abordagens de avaliação e análise) para serem empregadas durante a avaliação.

A realização de avaliações de risco inclui as seguintes tarefas específicas [NIST 800-30]:

- Identificar fontes de ameaças relevantes para a organização.
- Identificar os eventos de ameaça que poderiam ser produzidos por essas fontes.
- Identificar vulnerabilidades dentro da organização que possam ser exploradas por fontes de ameaças através de eventos específicos e condições predisponentes que poderiam afetar a exploração bem-sucedida.
- Determinar a probabilidade das fontes de ameaça identificadas iniciarem eventos específicos e a probabilidade de que estes eventos sejam bem-sucedidos.
- Determinar os impactos adversos às operações e bens organizacionais, aos indivíduos, a outras organizações e à nação resultante da exploração de vulnerabilidades por ameaças.

A comunicação e a partilha das informações consistem nas seguintes tarefas específicas [NIST 800-30]:

- Comunicar os resultados da avaliação de risco.
- Compartilhar informações desenvolvidas na execução da avaliação de risco para apoiar outras atividades de gerenciamento de risco.

1.2 Políticas e procedimentos de segurança da informação

1.2.1 Compreendendo as políticas e procedimentos de segurança

É comum que as políticas de segurança da informação variem entre as organizações com base no modelo de negócio, em indústrias específicas e nos riscos de segurança únicos enfrentados pela organização. Mesmo com uma ampla variação, os objetivos das políticas de segurança são semelhantes. A base de todas as políticas de segurança deve ser uma avaliação de risco de segurança que examina ameaças específicas e como elas afetam a organização. [Jackson, 2010]

Exemplos de políticas de segurança incluem, mas não estão limitados a [Jackson, 2010]:

Uso aceitável: Esta política define práticas que um usuário de um sistema de computador deve aderir para ser compatível com as políticas e procedimentos de segurança da organização. Esta política abrange comportamentos aceitáveis e não aceitáveis no uso de recursos digitais, como redes, sites e dados. Além disso, a política pode ser aplicada a usuários internos e externos dos sistemas de uma organização. É importante que os usuários do sistema compreendam e sigam a política em todos os momentos. Para evitar confusões e violações acidentais da política, deve definir regras específicas relativas a comportamento aceitável, comportamento inaceitável e comportamento exigido.

Acesso mínimo: Esta política define os níveis mínimos de acesso necessários para executar determinadas tarefas. O objetivo desta política é impedir que as pessoas recebam direitos de acesso superiores ao necessário para executar suas tarefas. Ter direitos de acesso mais altos do que o necessário pode fornecer oportunidades para abuso inadvertido ou intencional de privilégios de usuário.

Acesso à rede: Esta política define critérios para acessar vários tipos de redes, como redes de área local (LANs) e redes sem fio. Além disso, esta política pode definir o que é permitido ou não enquanto estiver na rede. Esta política, muitas vezes, proíbe os usuários de adicionar dispositivos não autorizados, como roteadores e hot spots na rede.

Acesso remoto: Esta política requer o que é necessário para que o acesso remoto à rede possa ser concedido tanto a funcionários internos como a externos (não-funcionários). O uso de VPN é muitas vezes coberto nesta política.

Acesso à internet: Esta política define o uso permitido da Internet pelos funcionários e convidados de uma organização. O escopo desta política inclui os tipos de sites que podem ou não serem acessados, como sites de jogos de azar ou pornografia, e aborda se a utilização não comercial da Internet é permitida. Embora alguns dos itens cobertos na política também possam ser abordados na política de uso aceitável, algumas organizações optam por definir essa política separadamente por causa do número de pessoas que fazem negócios pela Internet.

Gerenciamento de conta de usuário: Esta política define a criação, manutenção e exclusão de contas de usuário. A auditoria regular das contas de usuário também está coberta nesta política para garantir seu cumprimento.

Classificação de dados: Há muitas maneiras de classificar dados de uma perspectiva de segurança. Neste programa, o termo "dados sigilosos" é usado como um termo geral para quaisquer dados que devem ser protegidos para evitar perda. Uma política de classificação de dados define os diferentes tipos de dados que são considerados sigilosos e devem ser protegidos. Por ter uma política de classificação de dados, uma organização pode criar controles para proteger os dados com base em seu valor para a organização e seus clientes. Normalmente a área de negócios que cria os dados é responsável por sua classificação baseada em uma estrutura de classificação padrão.

O seguinte é um exemplo de estrutura de classificação de dados (a partir de um contexto de negócios):

- **Público**: qualquer pessoa, dentro ou fora da organização, pode visualizar esses dados (por exemplo, documentos públicos e páginas Web).
- **Confidencial**: esta é normalmente a classificação padrão para todos os documentos criados internamente. Esses documentos podem incluir e-mails, relatórios e apresentações que são usados internamente na organização. Um exemplo disto seria um relatório de vendas. Apenas os usuários autorizados a acessar estes dados podem trabalhar com este nível de informação. Contratos de não divulgação são muitas vezes necessários antes de compartilhar esse tipo de informação com terceiros, como consultores.
- **Altamente confidencial**: este é um nível mais elevado de confidencialidade para informações sigilosas que devem estar disponíveis apenas para certas pessoas na organização. Isso inclui informações como segredos comerciais, planos estratégicos, projetos de produtos e dados financeiros não-públicos. O compartilhamento deste tipo de dados não é permitido exceto com permissão explícita do proprietário dos dados.
- **Privado**: trata-se de informações que são frequentemente restritas aos funcionários da organização que devem estar especificamente autorizados a ter acesso a eles. Se divulgadas, essas informações podem ter um impacto negativo importante como danos financeiros para a organização. Devido ao alto risco associado à perda, a informação privada deve ser protegida com extremo cuidado. Esses dados podem incluir informações de pesquisa e desenvolvimento, planos de fusão e aquisição, bem como informações de clientes, como informações de cartão de crédito e de conta.
- **Segredo**: no contexto corporativo, trata-se de informações que uma organização recebe de uma parte externa para fazer alterações, mas que não é permitido tornar-se conhecido dentro ou fora da organização. Um exemplo no contexto corporativo seria um documento de projeto criado por um consultor trabalhando em um novo tipo de tecnologia que envolve a colaboração de outras empresas, cada uma das quais deve manter as informações em um nível secreto até que a tecnologia esteja pronta para ser revelada. É comparável a “altamente confidencial” com a diferença que pode não ter nenhum valor tangível para a própria organização. A este respeito, é diferente de um segredo comercial. No entanto, a divulgação de informações secretas pode causar danos à organização, outras organizações ou ao país. No contexto militar e governamental, trata-se de informações que podem ser desenvolvidas ou obtidas, mas devem ser conhecidas apenas por pessoas com certos níveis de habilitação de segurança. No contexto militar, incluiria dados de projetos científicos ou de pesquisa

que incorporassem novos desenvolvimentos tecnológicos ou técnicas com aplicações militares de importância vital para a defesa nacional.

Gerenciamento de configuração e mudança: Esta política pode ter um contexto operacional normal, como descrever como as alterações aos sistemas são gerenciadas e configuradas, a fim de evitar interrupções devido a impactos inesperados. Do ponto de vista de segurança, o gerenciamento de configuração controla como as configurações de segurança são aplicadas aos dispositivos e aplicativos. O risco é que uma alteração não autorizada em um dispositivo seguro pode causar uma vulnerabilidade de segurança que pode ser detectada.

Outro risco é que uma alteração não autorizada no código ou na configuração do aplicativo pode criar uma vulnerabilidade de segurança. Essa política inclui padrões de configuração a serem usadas, um processo de aprovação para todas as alterações e um processo de reversão se ocorrerem problemas. Esta política pode ser aplicada a todos os serviços de TI, aplicativos e dispositivos em uma organização.

Segurança do servidor: Esta política confere responsabilidade ao(s) proprietário(s) do servidor para seguir práticas de segurança corporativa, bem como práticas recomendadas da indústria para: instalar, configurar e operar servidores e sistemas. Além disso, as configurações de *baseline* devem ser definidas e mantidas. Exemplos de práticas descritas nesta política incluem requisitos de segurança, backup e recuperação e limitação de serviços ativos necessários para executar aplicativos. Também incluso nessa política podem existir requisitos de monitoramento e auditoria para garantir que o servidor esteja configurado e atualizado corretamente.

Dispositivos móveis: Os dispositivos móveis têm um conjunto único de preocupações de segurança, portanto, uma política separada pode ser necessária apenas para dispositivos móveis. Por exemplo, laptops e telefones inteligentes podem ser facilmente perdidos ou roubados, o que pode resultar em perda de dados da empresa e/ou pessoais. Estes dispositivos também têm um alto risco de contato com *malwares*. Esses riscos exigem regras e precauções específicas que devem ser seguidas para mitigar os riscos e limitar a exposição da organização às ameaças de segurança. Esta política pode incluir requisitos para os quais os dados devem ser criptografados, a instalação e manutenção de versões atuais do software *anti-malware* e quando as senhas são necessárias para acessar o dispositivo. Além disso, os tipos de informações organizacionais que podem residir em dispositivos móveis são definidos nesta política. A segurança física também pode ser abordada, como ter bloqueios de cabo para computadores portáteis e ter procedimentos para relatar dispositivos perdidos ou roubados.

Acesso de convidado: Essa política define as práticas que devem ser implementadas para proteger a organização, permitindo que a empresa hospede convidados e outras pessoas em redes organizacionais. Um aspecto desta política é exigir que os hóspedes leiam e aceitem políticas de uso aceitáveis antes de conceder acesso à rede. Essa política pode ser implementada de várias maneiras, como fazer com que os clientes assinem uma política de uso aceitável, e fornecer a eles um código para acesso temporário. A principal intenção desta política é reforçar os padrões de segurança da organização e ainda fornecer procedimentos para permitir que os hóspedes acessem a rede ou a Internet.

Segurança física: Esta política define os controles necessários para instalações físicas, uma vez que estar em proximidade física com dispositivos seguros pode aumentar o risco de violação de segurança. Esta política também pode cobrir outros riscos, tais como perda de energia, roubo, incêndio e desastres naturais. Também deve ser abordado quais os dispositivos podem ser retirados ou trazidos para a empresa, especialmente para as áreas que abrigam informações sigilosas.

Política de senhas: Esta política define os requisitos mínimos para senhas fortes e outras práticas seguras de senha, como o período permitido entre as alterações de senhas obrigatórias, como as pessoas protegem a privacidade de suas senhas (como não usar o recurso "lembrar senha", proibindo o compartilhamento de senhas e proibindo a transmissão de senhas por e-mail). Esta política pode ser aplicada a aplicativos, contas de usuário e outros locais onde as senhas são necessárias.

Proteção contra *malware*: Esta política define uma estrutura de defesas e comportamentos para prevenir, detectar e remover *malwares*. Uma vez que o *malware* e o *spyware* podem ser apanhados a partir de uma variedade de fontes, esta é uma política importante para que todos na organização compreendam e a sigam. Por exemplo, esta política pode restringir o uso de unidades USB.

Resposta a incidentes: Esta política descreve como responder a um incidente relacionado à segurança. Esses incidentes podem variar desde a descoberta de *malwares* e violações da política de uso aceitável até o acesso não autorizado a dados confidenciais. É importante que esta política seja implementada antes que ocorra um incidente para evitar ter que determinar as respostas apropriadas caso a caso. Esta política também aborda a comunicação, incluindo as respostas dos meios de comunicação e notificação de aplicação da lei.

Política de auditoria: Esta política autoriza os auditores a solicitar acesso à sistemas com a finalidade de realizar uma auditoria. A equipe de auditoria pode precisar de acesso aos dados de *log*, registros de tráfego de rede e outros dados forenses.

Licenciamento de software: Esta política aborda como a organização obtém e licencia o software que utiliza. Se as licenças de software comercial forem violadas, a organização corre o risco de multas e ações legais. Devido a isso, é importante que as licenças sejam identificadas e monitoradas. Fazer o download e instalar software não aprovado é uma proibição fundamental frequentemente encontrada nesta política.

Monitoramento eletrônico e privacidade: As organizações têm o direito e a responsabilidade de monitorar as comunicações eletrônicas em todo o hardware e recursos da empresa. Isso inclui correspondência de e-mail e mídias sociais. Esta política descreve qual o monitoramento é realizado pela organização e quais dados estão sujeitos à coleta. Leis variam entre os países, por isso é necessário aconselhamento jurídico antes de escrever esta política. [Jackson, 2010]

1.2.1.1 Procedimentos de segurança

Os procedimentos de segurança especificam as etapas a serem tomadas na implementação de uma política ou controle específico e as etapas a serem tomadas em resposta a um incidente de segurança. Procedimentos formais e documentados facilitam a implementação das políticas de segurança e controles obrigatórios.

As políticas, normas e diretrizes descrevem os controles de segurança que devem estar em vigor, enquanto um procedimento descreve detalhes específicos, explicando como implementar os controles de segurança passo a passo. Por exemplo, um procedimento poderia ser escrito para explicar como conceder níveis de acesso de usuário, detalhando cada passo que precisa ser tomado para garantir o nível correto de acesso é concedido para que os direitos de usuário satisfaçam a política, normas e diretrizes aplicáveis.

1.2.2 Análise de políticas e procedimentos de segurança

Antes de avaliar um conjunto de políticas e procedimentos de segurança, é importante determinar os objetivos da avaliação e definir um conjunto de critérios para avaliar a adequação das políticas e procedimentos. Em alguns casos, os critérios podem ser definidos por padrões como COBIT [COBIT], ISO27001 [ISO27001] ou PCI [PCI].

Além disso, é necessário definir:

- Que recursos são necessários em termos de competências e conhecimentos em áreas específicas a serem avaliadas.
- Como medir a adequação das políticas e procedimentos.
- O que medir e avaliar (por exemplo, eficácia, eficiência, usabilidade, adoção).
- Onde as políticas e procedimentos estão localizados na organização.
- Uma lista de verificação para orientar a avaliação.

A lista de verificação funciona como um guia que direciona o auditor onde procurar e o que esperar. Ferramentas, como de auditoria de senha, podem ser úteis no teste de determinados controles para avaliar se estão cumprindo seus objetivos e gerando informações que possam ser usadas posteriormente na avaliação de riscos. O auditor analisa a conformidade da "prova" às políticas, aos controles e às normas.

Algumas das tarefas na lista a seguir são de natureza estática, enquanto outras, como a observação de processos em ação, são dinâmicas. O auditor faz o seguinte:

- Revisar a documentação do sistema.
- Entrevistar as pessoas sobre a sua percepção da eficácia das políticas e procedimentos.
- Entrevistar o pessoal-chave envolvido nos processos que estão sendo controlados.
- Verificar sistemas e processos acessados por visitantes.
- Analisar resultados de auditoria anteriores para descobrir tendências
- Analisar *logs* e relatórios
- Revisar a configuração do controle técnico, como configuração do firewall e configuração do sistema de detecção de intrusão
- Analisar amostras de transações de dados para quaisquer anomalias ou transações suspeitas [Jackson, 2010]

1.2.2.1 Controles

Os controles de segurança são salvaguardas técnicas ou administrativas ou contramedidas para evitar, neutralizar ou minimizar a perda ou indisponibilidade devido às ameaças que agem em sua vulnerabilidade correspondente, ou seja, risco de segurança [Northcutt, 2014]. Por exemplo, um controle de segurança em um sistema de folha de pagamento pode ser que duas pessoas devem aprovar separadamente uma alteração nas informações de remuneração de um empregado. Testadores em segurança devem estar cientes dos controles específicos em sua organização e incluir testes para eles nos testes de segurança.

Os principais tipos de controle de segurança são administrativos, técnicos e físicos. Em cada categoria, os controles específicos que podem ser implementados são preventivos, detectives, corretivos ou de recuperação. Esses tipos de controle funcionam em conjunto e, em geral, os controles devem ser fornecidos de cada categoria para proteger efetivamente um ativo. [Jackson, 2010]

Uma lista dos 20 principais controles críticos de segurança pode ser encontrada em www.sans.org. [Web-1]

1.2.2.2 Testes de segurança

A principal diferença nos testes de segurança em comparação com uma análise estática de políticas e procedimentos de segurança é a utilização dos resultados de testes, concebidos especificamente para verificar ou validar a eficácia das políticas e procedimentos de segurança. Esses testes se concentram no risco de que uma política de segurança pode estar em vigor, pode ser seguida, mas não é eficaz na proteção de ativos.

Também é possível ser informado durante a execução das avaliações das políticas e procedimentos de segurança que determinadas tarefas são executadas. Um teste de segurança dessas tarefas pode ajudar a determinar a eficácia das políticas e procedimentos de segurança na prática. Por exemplo, uma política e procedimento de senha pode parecer razoável e eficaz no papel, mas quando uma ferramenta de quebra de senhas é usada, o procedimento pode ficar aquém de seus objetivos.

As políticas e procedimentos de segurança podem ser uma fonte de testes de segurança, no entanto, o testador em segurança deve ter em mente que os ataques estão sempre evoluindo. Novos ataques surgem e, assim como em qualquer aplicativo de software, novos defeitos podem tornar-se evidente - todos os quais são razões para realizar testes de segurança a partir da mentalidade atacante.

1.3 A auditoria de segurança e seu papel no teste de segurança

A auditoria de segurança é um exame e uma avaliação manual que identificam deficiências nos processos e na infraestrutura de segurança de uma organização. As auditorias de segurança no nível processual (por exemplo, para rever os controles internos) podem ser realizadas manualmente. As auditorias de segurança no nível arquitetural são muitas vezes realizadas com ferramentas de auditoria de segurança, que podem ser alinhadas com uma solução de fornecedor específica para redes, servidores e estações de trabalho.

Assim como os testes de segurança, uma auditoria de segurança não garante que todas as vulnerabilidades serão encontradas. No entanto, a auditoria é mais uma atividade no processo de segurança para identificar áreas problemáticas e indicar onde a remediação é necessária.

Em algumas abordagens de auditoria de segurança, o teste é executado como parte do processo de auditoria. No entanto, o escopo da auditoria de segurança é muito maior do que os testes de segurança. A auditoria de segurança geralmente investiga áreas como procedimentos, políticas e controles que são difíceis de testar de forma direta. Os testes de segurança estão mais envolvidos com as tecnologias de suporte à segurança, como configuração de firewall, aplicação correta de autenticação, criptografia e de direitos de usuário.

Existem cinco pilares para a auditoria de segurança [Jackson, 2010]:

Avaliação: As avaliações documentam e identificam ameaças potenciais, ativos-chave, políticas e procedimentos e tolerância da gerência ao risco. As avaliações não são eventos únicos. Uma vez que o ambiente e as empresas estão constantemente em fluxo, avaliações devem ser realizadas em uma base regular. Isso também fornece a oportunidade de saber se as políticas de segurança ainda são relevantes e eficazes.

Prevenção: Isso vai além da tecnologia e inclui controles administrativos, operacionais e técnicos. A prevenção não é realizada apenas através da tecnologia, mas também através de políticas, procedimentos e conscientização. Embora a prevenção de todos e quaisquer ataques não seja realista, a combinação de defesas pode ajudar a tornar muito mais difícil para um atacante ter sucesso.

Detecção: Detecção é como uma violação de segurança ou intrusão é identificada. Sem mecanismos de detecção adequados, existe o risco de não saber se a rede foi comprometida. Controles espiões ajudam a identificar incidentes de segurança e fornecem visibilidade em atividades na rede. A detecção precoce de incidentes permite uma reação adequada para recuperar serviços rapidamente.

Reação: O tempo de reação é muito reduzido com boas defesas de segurança e mecanismos de detecção. Embora as violações de segurança sejam más notícias, é importante saber se ocorreu uma. O tempo de reação rápida é crítico para minimizar a exposição ao incidente. A reação rápida requer boas defesas preventivas e mecanismos de detecção para fornecer os dados e o contexto necessário para a resposta. A velocidade e a eficiência da resposta a incidentes são indicadores-chave da eficácia dos esforços de segurança de uma organização.

Recuperação: A recuperação começa com a determinação do que ocorreu para que os sistemas possam ser recuperados sem recriar a mesma vulnerabilidade ou condição que causou o incidente, em primeiro lugar. A fase de recuperação não termina com a restauração do sistema. Há também a análise de causa raiz que determina quais mudanças precisam ser feitas para processos, procedimentos e tecnologias a fim de reduzir a probabilidade do mesmo tipo de vulnerabilidade no futuro. Um auditor deve assegurar-se de que a organização tenha um plano de recuperação que inclua formas de prevenir futuros incidentes semelhantes.

1.3.1 Finalidade de uma auditoria de segurança

O seguinte é uma lista de itens que podem ser detectados em uma auditoria de segurança:

- **Segurança física inadequada**: uma política de segurança pode exigir a criptografia de todos os dados do cliente, tanto em armazenamento quanto em transmissão. Por exemplo, durante a auditoria, descobre-se que uma vez por semana, um arquivo de informações do cliente é enviado por relatório físico a todos os gerentes. Este relatório é descartado todas as semanas, mas descobriu-se que alguns gerentes estão jogando os relatórios físicos no lixo, onde podem ser encontrados facilmente por qualquer pessoa que passe perto do local (ou seja, "dumpster diving").
- **Manutenção de senha inadequada**: uma política de segurança pode exigir que cada usuário altere sua senha a cada 30 dias. Uma auditoria de segurança revela que as senhas são alteradas, mas muitos usuários simplesmente alternam entre "PasswordA" e "PasswordB" a cada mês. (O histórico de senhas é uma característica comum nas ferramentas de auditoria de senhas.)
- **Controles inadequados para direitos de usuário e privilégios de compartilhamento**: um exemplo de uma conclusão negativa poderia ser quando os usuários tiveram mais direitos de acesso aos itens do que eles precisam para executar seu trabalho. Outro exemplo pode ser quando os arquivos de um usuário individual são compartilhados na rede quando devem ser privados. Esta é uma preocupação particular para os usuários com notebook e especialmente aqueles que acessam a intranet através de conexões Wi-Fi em casa ou em locais públicos.
- **Segurança inadequada no nível do servidor**: as áreas de auditoria específicas incluem:
  - Alocação de portas e segurança de acesso.
  - Proteção de dados.
  - Proteção de contas de usuário (*logins* e outras informações sigilosas).
- **Aplicação inadequada de atualizações de segurança de fornecedores.**
- **Mecanismos de detecção de intrusão inadequados.**
- **Planos de resposta inadequados em caso de violação de segurança.**

1.3.2 Identificação, avaliação e mitigação de riscos

Uma vez que as áreas problemáticas foram identificadas pela auditoria, o risco deve ser avaliado e um plano de melhoria deve ser implementado. O relatório do auditor pode incluir recomendações, bem como outras áreas de risco. A partir deste ponto, as atividades de identificação, avaliação e mitigação de riscos podem ser planejadas.

Identificação de risco é o processo de documentação de um risco ou uma área de risco. No contexto da segurança de TI, os riscos são relacionados à segurança. A avaliação de risco é a atividade que atribui um valor aos riscos identificados. É importante entender que os modelos tradicionais de avaliação de riscos não são suficientes para enfrentar os riscos de segurança. Qualquer modelo ou abordagem de avaliação de riscos de segurança deve ser orientado especificamente para perfis de risco de segurança de TI.

Riscos de segurança muitas vezes são medidos em termos de exposição ao risco. A exposição ao risco é calculada multiplicando o impacto ou perda potencial pela probabilidade de ocorrência dessa perda. Por exemplo, se as informações da conta de um cliente estiverem comprometidas, qual seria o impacto? E se esse cliente tivesse $ 100 milhões de ativos em depósito?

A probabilidade de ocorrência pode ser determinada pela aplicação de um modelo de avaliação de risco de segurança como o encontrado no Guia para Realização de Avaliações de Risco [NIST 800-30]. Outro excelente guia para realizar avaliações de risco de segurança é a Metodologia de Avaliação de Risco OWASP [OWASP2]. As informações a seguir são extraídas do [NIST 800-30].

Os modelos de risco definem os fatores de risco a serem avaliados e as relações entre eles. Os fatores de risco são características utilizadas em modelos de risco como insumos para determinar os níveis de risco nas avaliações de risco. Fatores de risco também são usados extensivamente em comunicações de risco para destacar o que afeta fortemente os níveis de risco em situações, circunstâncias ou contextos particulares.

Fatores de risco típicos incluem ameaça, vulnerabilidade, impacto, probabilidade e condição predisponente. Os fatores de risco podem ser decompostos em características mais detalhadas (por exemplo, ameaças decompostas em fontes de ameaças e eventos de ameaça). Essas definições são importantes para que as organizações documentem antes de realizar avaliações de risco porque as avaliações dependem de atributos bem definidos de ameaças, vulnerabilidades, impacto e outros fatores de risco para efetivamente determinar o risco.

1.3.2.1 Ameaças

Uma ameaça é qualquer circunstância ou evento com o potencial de afetar adversamente as operações organizacionais e os bens, indivíduos, outras organizações ou um país por meio de um sistema de informação por acesso não autorizado, destruição, divulgação ou modificação de informações e/ou negação de serviço.

Os eventos de ameaça são causados por fontes de ameaça. Uma fonte de ameaça é caracterizada como:

- A intenção e o método visando a exploração de uma vulnerabilidade.
- Uma situação e um método que podem explorar acidentalmente uma vulnerabilidade.

Em geral, os tipos de ameaças incluem:

- Ataques cibernéticos ou físicos hostis.
- Erros humanos de omissão ou autorização.
- Falhas estruturais de recursos controlados pela organização (por exemplo, hardware, software, controles ambientais)
- Desastres naturais, causados pelo homem, acidentes e falhas que escapam ao controle da organização.

Várias taxonomias de fontes de ameaças foram desenvolvidas. Algumas taxonomias usam o tipo de impactos adversos como um princípio organizador. Várias fontes de ameaças podem iniciar ou causar o mesmo evento de ameaça - por exemplo, um servidor de provisionamento pode ser desconectado por um ataque de negação de serviço, um ato deliberado por um administrador de sistema mal-intencionado, um erro administrativo, uma falha de hardware ou uma falha de hardware. Falha de energia.

1.3.2.2 Vulnerabilidades e condições predispostas

Uma vulnerabilidade é uma fraqueza em um sistema de informação, procedimentos de segurança do sistema, controles internos ou implementação que poderiam ser explorados por uma fonte de ameaça.

A maioria das vulnerabilidades do sistema de informação podem ser associadas a controles de segurança que não foram aplicados (intencionalmente ou não), ou que foram aplicados, mas mantêm alguma fraqueza. No entanto, também é importante permitir a possibilidade de vulnerabilidades emergentes que podem surgir naturalmente ao longo do tempo como missões organizacionais, evolução de funções de negócios, mudanças em ambientes de operação, novas tecnologias proliferam e novas ameaças surgem. No contexto de tais alterações, os controles de segurança existentes podem tornar-se inadequados e poderão ser reavaliados para efeitos de eficácia. A tendência de controles de segurança potencialmente degradarem a eficácia ao longo do tempo reforça a necessidade de manter avaliações de risco durante todo o ciclo de vida do software e, também, a importância dos programas de monitorização contínua para obter uma consciência situacional da postura de segurança organizacional.

Vulnerabilidades não são identificadas apenas dentro dos sistemas de informação. Vendo os sistemas de informação em um contexto mais amplo, as vulnerabilidades podem ser encontradas nas estruturas de governança organizacional (por exemplo, a falta de estratégias eficazes de gerenciamento de risco e enquadramento de risco adequado, comunicações interagências pobres, decisões inconsistentes sobre prioridades relativas de missões/funções empresariais ou desalinhamento da arquitetura corporativa para apoiar a missão/atividades de negócios).

Vulnerabilidades também podem ser encontradas em relações externas (i.e., dependências de fontes de energia específicas, cadeias de suprimento, tecnologias de informação e provedores de telecomunicações), processos de missão/negócios (por exemplo, processos mal definidos que não estão conscientes do risco), arquiteturas de segurança da informação (por exemplo, decisões arquitetônicas pobres resultando em falta de diversidade ou resiliência em sistemas de informação organizacional).

1.3.2.3 Impacto

O nível de impacto de um evento de ameaça é a magnitude do dano que pode ser esperado resultar das consequências de divulgação, modificação, destruição não autorizadas de informações, ou da perda de informações ou disponibilidade do sistema de informação. Esse dano pode ser vivenciado por uma variedade de setores organizacionais e não-organizacionais, incluindo:

- Líderes.
- Proprietários de negócios.
- Administradores de informação.
- Proprietários de processo de processos de negócios.
- Proprietários de sistemas de informação.
- Indivíduos ou grupos do setor público ou privado dependentes da organização - em essência, qualquer pessoa com um interesse pessoal nas operações, bens ou indivíduos da organização, incluindo outras organizações em parceria com a organização ou um país.

As seguintes informações devem ser explicitamente documentadas por uma organização:

- O processo usado para conduzir determinações de impacto
- Premissas relacionadas a determinações de impacto
- Fontes e métodos para obter informações sobre o impacto
- A justificativa para as conclusões alcançadas em relação às determinações de impacto

As organizações podem definir explicitamente como prioridades e valores estabelecidos orientam a identificação de ativos de alto valor e os potenciais impactos adversos para os atores organizacionais. Se tais informações não forem definidas, as prioridades e os valores relacionados à identificação de alvos de fontes de ameaça e impactos organizacionais associados podem normalmente ser derivados do planejamento estratégico e políticas. Por exemplo, os níveis de categorização de segurança indicam os impactos organizacionais de comprometer diferentes tipos de informação.

1.3.2.4 Probabilidade

A probabilidade de ocorrência aborda a probabilidade (ou possibilidade) de que o evento de ameaça resulte em um impacto adverso, independentemente da magnitude do dano que pode ser esperado. Este é um fator de risco ponderado baseado em uma análise da probabilidade de que uma determinada ameaça é capaz de explorar uma determinada vulnerabilidade (ou conjunto de vulnerabilidades). O fator de risco de probabilidade combina uma estimativa da probabilidade de que o evento de ameaça será iniciado com uma estimativa da probabilidade de impacto (ou seja, a probabilidade de que o evento de ameaça resulte em impactos adversos).

Para as ameaças adversas, uma avaliação da probabilidade de ocorrência normalmente se baseia em:

- Intenção
- Capacidade
- Segmentação

Para eventos que não sejam adversos, a probabilidade de ocorrência é estimada usando evidências históricas, dados empíricos ou outros fatores. Observe que a probabilidade de que um evento de ameaça será iniciado

ou ocorrerá é avaliado em relação a um período específico (por exemplo, nos próximos seis meses, no próximo ano ou no período até que um marco especificado seja atingido).

Se um evento de ameaça é quase certo para ser iniciado ou ocorrer no prazo (especificado ou implícito), a avaliação de risco pode levar em consideração a frequência estimada do evento. A probabilidade de ocorrência de ameaças também pode ser baseada no estado da organização (incluindo, por exemplo, sua missão básica, processos de negócios, arquitetura empresarial, arquitetura de segurança da informação, sistemas de informação e ambientes nos quais esses sistemas operam. A presença e a eficácia dos controles de segurança implantados para proteger contra o comportamento não autorizado ou indesejável, detectar e limitar os danos, e/ou manter ou restaurar as capacidades da missão ou negócio também devem ser levados em consideração.

1.3.2.5 Determinação do nível de risco de segurança

A probabilidade de avaliação de ocorrência e a avaliação de impacto podem ser combinadas para calcular uma gravidade geral para o risco. As pontuações de avaliação específicas podem ser utilizadas como base para a conclusão da matriz de risco. Em outros casos, estimativas (baixa, média ou alta) podem ser usadas.

A pontuação para a matriz de risco pode ser baseada numa escala de 0 a 9, onde os valores numéricos são determinados por critérios específicos. Por exemplo, os critérios de probabilidade de risco poderiam ser avaliados quanto à privacidade dos dados como:

- **Baixo (0 < 3)**: Os dados privados não são armazenados em dispositivos locais e são criptografados quando armazenados em mídia segura.
- **Média (3 < 6)**: Os dados privados podem residir em dispositivos como computadores portáteis, mas são criptografados.
- **Alta (6 < 9)**: Não se sabe exatamente se os dados particulares residem em dispositivos locais. A criptografia não pode ser assegurada.

Da mesma forma, os critérios de impacto de risco poderiam ser avaliados na mesma escala de 0 a 9 com base em critérios específicos. Por exemplo:

- **Baixo (0 < 3)**: Dados privados comprometidos impactariam menos de 200 pessoas.
- **Médio (3 < 6)**: Dados privados comprometidos impactariam entre 200 e 1.000 pessoas.
- **Alto (6 < 9)**: Dados privados comprometidos impactariam mais de 1.000 pessoas.

No entanto, o testador chega às estimativas de probabilidade e impacto, combinando-as em uma classificação de severidade final para o item de risco. Se houver boas informações de impacto nos negócios, isso deve ser usado em vez das informações de impacto técnico. Se não houver nenhuma informação sobre o negócio, então o impacto técnico é a melhor opção.

Abaixo está uma amostra de uma matriz de risco que pode ser usada para determinar a gravidade dos riscos individuais.

Gravidade Geral do Risco		Probabilidade		
		Baixo	Médio	Alto
Impacto do Risco	Alto	Médio	Alto	Crítico
	Médio	Médio	Médio	Alto
	Baixo	Baixo	Baixo	Médio

Na matriz de exemplo acima, se a probabilidade é média e o impacto é alto, a gravidade geral é alta.

Além disso, o relatório de avaliação de risco deve identificar se o risco está em curso. Riscos contínuos indicam uma maior probabilidade de ocorrência de perda.

A gravidade de um risco determina a importância relativa da mitigação do risco. Quanto maior a gravidade do risco, mais imediata a resposta deve ser exigida. O nível de pormenor fornecido em qualquer avaliação de risco em particular é consistente com a finalidade da avaliação de risco e o tipo de insumo necessário para apoiar a determinação de verossimilhança e impacto subsequente.

1.3.3 Pessoas, processos e tecnologia

Há também três componentes para as práticas de TI de uma organização: pessoas, processos e tecnologia. Todos estes têm um impacto na segurança. De acordo com Chris Jackson em seu livro, *Network Security Auditing* [Jackson, 2010], "*Todos os incidentes de segurança, para registros de clientes perdidos, geralmente podem ser rastreados até uma deficiência que pode ser atribuída a pessoas, processos ou tecnologia.*"

Pessoas: as pessoas podem incluir usuários finais, administradores de sistema, proprietários de dados e gerentes da organização. Cada pessoa tem diferentes níveis de habilidades, atitudes e agendas, o que influencia a forma como a segurança as afeta e como elas afetam a eficácia dos controles de segurança. Independentemente da presença de políticas de segurança, procedimentos e controles, eles serão ineficazes se as pessoas não os seguir. Se as pessoas não seguem as políticas de segurança, pode haver uma necessidade de esforço de remediação, como a necessidade de treinamento, conscientização de segurança ou penalidades por não-conformidade. As estruturas organizacionais e as políticas de segurança são muitas vezes dirigidas por pessoas, tanto internas como externas em uma organização.

Processo: os processos definem como os serviços de TI, incluindo os serviços relacionados à segurança, são entregues. Em um contexto de segurança, os processos incluem os procedimentos e padrões que são colocados em prática para proteger ativos valiosos. Para serem eficazes, os processos devem ser definidos, atualizados, consistentes e seguir as melhores práticas de segurança. Os processos definem funções e responsabilidades, controles, ferramentas e etapas específicas envolvidas na execução de uma tarefa.

Tecnologia: A tecnologia engloba as instalações, equipamentos, hardware e software que automatizam ou suportam um negócio. A tecnologia permite que as pessoas realizem trabalhos repetitivos mais rápido e com menos erro do que se executado manualmente sem ele. Na verdade, algumas tarefas como a imposição de senha seriam impossíveis sem as ferramentas certas. O risco é que a tecnologia usada incorretamente pode ajudar as pessoas a cometerem erros mais rapidamente.

Estas três áreas podem ser pensadas como um "triângulo de ferro", que juntos formam uma solução completa de TI. Se qualquer uma das três áreas são ignoradas, toda a entrega de TI e esforço de segurança sofre.

Ao avaliar os controles de segurança, o auditor deve olhar para um sistema na perspectiva de um invasor e antecipar como as pessoas, processos ou tecnologias poderiam ser exploradas para obter acesso não autorizado aos ativos valiosos.

A gestão em uma organização é muitas vezes surpreendida nos mecanismos de segurança que pensavam ser seguros, mas não os são. A única maneira de saber com certeza se uma defesa de segurança específica está funcionando eficazmente é testando o sistema de uma perspectiva do invasor. Isso é muitas vezes conhecido como *hacking* ético ou teste de penetração.

É aqui que a relação entre a auditoria e o teste se torna mais direta. A auditoria identifica deficiências e áreas de importância para testar. Testes de segurança são o meio pelo qual é provado ou desmentido que os controles de segurança estão realmente no lugar e trabalhando efetivamente.

Exemplo de cenário: A agência fiscal de um país é objeto de uma auditoria de segurança. Uma das conclusões da auditoria é ser possível para os criminosos apresentar uma declaração fiscal fraudulenta e obter os reembolsos de imposto devido ao contribuinte. Este achado de auditoria é confirmado com testes de

segurança e o risco é classificado como "crítico". A agência fiscal reconhece a possibilidade de tal risco fraudulento, mas decide não agir sobre o risco até o ano seguinte.

Contribuintes fraudados, que seguiram todos os procedimentos de segurança prescritos, podem apresentar uma reclamação com a agência fiscal que estava ciente do defeito no processo de arquivamento de imposto. Neste caso, a agência fiscal seria responsável pela fraude.

2 Testes de segurança, objetivos, metas e estratégias (130 min)

Palavras-Chave

cross-site scripting, mascaramento de dados, *denial of service*, garantia da informação, política de segurança, teste de segurança, vulnerabilidade de segurança, ciclo de vida de software, estratégia de teste.

Objetivos de aprendizagem

2.1 Introdução

Sem objetivos de aprendizado para esta seção.

2.2 O Propósito dos testes de segurança

AS-2.2.1 (K2): Entenda por que os testes de segurança são necessários em uma organização, incluindo benefícios como redução de risco e níveis mais altos de confiança.

2.3 O Contexto organizacional

AS-2.3.1 (K2): Entender como as realidades do projeto, as restrições de negócio, o ciclo de vida do desenvolvimento de software e outras considerações afetam a missão da equipe de testes de segurança.

2.4 Objetivos do teste de segurança

AS-2.4.1 (K2): Explicar por que os objetivos dos testes de segurança devem estar alinhados com a política de segurança da organização e com outros objetivos de teste na organização

AS-2.4.2 (K3): Demonstrar a capacidade de identificar objetivos de teste de segurança com base em funcionalidade, atributos de tecnologia e vulnerabilidades conhecidas para um determinado cenário de projeto.

AS-2.4.3 (K2): Compreender a relação entre a garantia de informação e os testes de segurança.

2.5 Âmbito e cobertura dos objetivos de testes de segurança

AS-2.5.1 (K3): Para um determinado projeto, demonstrar a capacidade de definir a relação entre os objetivos do teste de segurança e a necessidade de resistência da integridade de ativos digitais e físicos sigilosos.

2.6 Abordagens de testes de segurança

AS-2.6.1 (K4): Analisar uma situação e determinar quais abordagens de teste de segurança têm maior probabilidade de sucesso.

AS-2.6.2 (K4): Analisar uma situação em que uma tentativa de teste de segurança falhou, identificando as prováveis causas da falha.

AS-2.6.3 (K3): Para um determinado cenário, demonstrar a capacidade de identificar as várias partes interessadas e ilustrar os benefícios dos testes de segurança para cada grupo de partes interessadas.

2.7 Melhorar as práticas de testes de segurança

AS-2.7.1 (K4): Analisar KPIs (indicadores-chave de desempenho) para identificar as práticas de testes de segurança que necessitam de aperfeiçoamento e elementos que não necessitam de melhorias.

2.1 Introdução

Antes de aplicar técnicas especializadas de teste de segurança, é importante entender o contexto mais amplo dos testes de segurança e seu papel dentro de uma determinada organização. Este entendimento responde às seguintes perguntas:

- Por que são necessários os testes de segurança?
- Qual é a finalidade dos testes de segurança?
- Como os testes de segurança se encaixam na organização?

Testes de segurança diferem de outras formas de testes funcionais em duas áreas significativas [BSTQB_ATTA_SYL]:

1. Técnicas-padrão para a seleção de dados de entrada dos testes pode não atender importantes problemas de segurança.
2. Os sintomas dos defeitos de segurança são muito diferentes dos encontrados com outros tipos de testes funcionais.

Os testes de segurança avaliam a vulnerabilidade de um sistema a ameaças ao tentar comprometer a política de segurança do sistema. O seguinte é uma lista de ameaças potenciais, que devem ser exploradas durante o teste de segurança [BSTQB_ATTA_SYL]:

- Cópia não autorizada de aplicativos ou dados.
- Controle de acesso não autorizado (por exemplo, capacidade de executar tarefas para as quais o usuário não tem direitos). Direitos de usuário, acesso e privilégios são o foco deste teste. Essas informações devem estar disponíveis nas especificações do sistema.
- Software que exibe efeitos colaterais não intencionais ao executar a função pretendida. Por exemplo, um reprodutor de mídia que reproduz corretamente áudio, mas faz isso, escrevendo arquivos para armazenamento temporário não criptografado exibe um efeito colateral, que pode ser explorado por piratas de software.
- Código inserido em uma página da Web que pode ser acessado por outros usuários (*cross-site scripting* ou XSS). Este código pode ser malicioso.
- Estouro de memória intermediária (saturação de memória intermediária) que pode ser causado por introduzir cadeias de dados num campo de entrada de interface de usuário, que são mais do que o código pode processar corretamente. Uma vulnerabilidade de estouro de memória representa uma oportunidade para executar instruções de código malicioso.
- *Denial of Service* (DOS), que impede que os usuários interajam com um aplicativo (por exemplo, sobrecarregando um servidor web com solicitações "desagradáveis").
- A intercepção, imitação ou alteração, e subsequente retransmissão de comunicações (por exemplo, transações de cartão de crédito) por um terceiro de tal forma que um usuário permanece inconsciente da presença desse terceiro (ataque "*Man in the Middle*").
- Quebrar os códigos de criptografia usados para proteger dados confidenciais.
- Bombas lógicas (às vezes chamadas de Ovos de Páscoa), que podem ser inseridas maliciosamente no código e que se ativam somente sob certas condições (por exemplo, em uma data específica). Quando as bombas lógicas são ativadas, elas podem executar ações maliciosas, como a exclusão de arquivos ou a formatação de discos.

Os testes de segurança devem ser integrados com todas as outras atividades de desenvolvimento e teste. Isso requer considerar as necessidades exclusivas da organização, quaisquer políticas de segurança existentes, conjuntos de habilidades atuais de testes de segurança e quaisquer estratégias de teste existentes.

2.2 O propósito dos testes de segurança

Como os testes de software em geral, os testes de segurança não podem garantir que um sistema ou organização fique a salvo do ataque. No entanto, os testes de segurança podem ajudar a identificar riscos e avaliar a eficácia das defesas de segurança existentes. Existem outras atividades para complementar testes de segurança, como auditorias e revisões de práticas de segurança.

Testes de segurança também mostram que a devida diligência tem sido realizada na proteção de ativos digitais. Em caso de violação de segurança pode resultar em uma ação legal. Se uma empresa pode mostrar que tomou medidas razoáveis para proteger seus ativos digitais com testes de vulnerabilidades, haverá o suficiente para uma defesa em um tribunal. Os testes de segurança também podem ser uma garantia para os clientes de que a organização toma medidas adequadas para proteger informações confidenciais.

2.3 O contexto organizacional

Segurança é muitas vezes um tipo de teste funcional realizado juntamente com outros tipos de testes. Com apenas uma determinada quantidade de tempo disponível para testes, um gerente de teste deve decidir quantos testes podem ser realizados, incluindo testes de segurança. Não é raro que os testes de segurança sejam considerados papel de especialistas e, portanto, terceirizados para uma organização especializada em testes de segurança. A extensão dos testes de segurança é, em última instância, impulsionada por riscos empresariais ou organizacionais baseados na segurança. Quando os riscos de segurança são numerosos em uma organização, testes de segurança mais extensos são necessários.

Como o teste de software, a segurança da informação é uma atividade do ciclo de vida. As necessidades de segurança devem ser definidas nas exigências, expressas em modelagem e implementadas em código. Em seguida, os testes de segurança podem verificar e validar a correção e a eficácia da implementação de segurança. A segurança não pode ser efetivamente corrigida em código ou testada em código. Somente quando a segurança é construída em software usando codificação segura e técnicas de modelagem pode software ser seguro.

As realidades de tempo limitado, recursos e escopo, juntamente com os níveis de risco, conjuntos de habilidades de teste de segurança e abordagens de ciclo de vida têm grande impacto no sucesso de uma equipe de testes de segurança em uma organização.

2.4 Objetivos do teste de segurança

2.4.1 O alinhamento dos objetivos de testes de segurança

A política de testes de segurança pode ser escrita uma vez que a política de segurança da organização tenha sido aprovada pela alta administração. É importante que os objetivos dos testes de segurança, conforme expressos na política, estejam alinhados com a política de segurança geral da organização. Caso contrário, serão realizados testes de segurança não autorizados ou os testes de segurança poderão não atingir os objetivos pretendidos.

2.4.2 Identificação dos objetivos do teste de segurança

Os objetivos do teste de segurança podem ser considerados da mesma forma que os objetivos dos testes funcionais, mas estão focados na segurança. Deve haver um ou mais objetivos de teste de segurança para cada recurso de segurança do sistema ou aplicativo.

Os objetivos do teste de segurança também devem basear-se em atributos de tecnologia (por exemplo, web, mobile, cloud, LAN) e vulnerabilidades conhecidas, tanto nas de aplicativo quanto genéricas. Por exemplo, os objetivos do teste de segurança podem incluir:

- Verificar se a autenticação de senha aplica a regra correta para a força da senha.
- Verificar se todos os campos de entrada de dados são validados para evitar ataques de *SQL injection.*
- Verificar se os arquivos de dados do cliente são criptografados com a força correta.

2.4.3 A diferença entre a garantia da informação e o teste de segurança

A Garantia da Informação é definida como: "*Medidas que protegem e defendem os sistemas de informação e a informação, assegurando sua disponibilidade, integridade, autenticação, confidencialidade e não-repúdio. Estas medidas incluem o reestabelecimento de sistemas de informação através da incorporação de capacidades de proteção, detecção e reação*". [NISTIR 7298]

O teste de segurança é "*um processo utilizado para determinar se os recursos de segurança de um sistema são implementados conforme projetados e se são adequados para o ambiente proposto do aplicativo*". [MDA1]

O termo Garantia da Informação é mais amplo e abrangente do que o termo Teste de Segurança. Esta relação é semelhante àquela entre Qualidade de Software e Teste de Software.

2.5 Âmbito e cobertura dos objetivos de testes de segurança

Quanto maior a necessidade de integridade de ativos digitais e físicos, maior a necessidade de cobertura dos objetivos do teste de segurança. Os objetivos do teste de segurança descrevem essencialmente o escopo dos testes de segurança. Se o escopo for muito pequeno, a confiança de que a segurança é adequada não será alcançada. Se o escopo for muito grande, os recursos podem ser esgotados antes que o teste possa ser concluído.

Os objetivos dos testes de segurança devem descrever quais testes de segurança se espera alcançar em relação à verificação e validação das proteções existentes para ativos digitais e físicos. Os objetivos dos testes de segurança devem se relacionar diretamente com ativos específicos, medidas de proteção, riscos e identificação de vulnerabilidades de segurança.

2.6 Abordagens de testes de segurança

A estratégia de teste de segurança é definida para formalizar e comunicar a direção geral de uma organização dos testes de segurança. As abordagens que implementam a estratégia de testes de segurança são então definidas.

2.6.1 Análise de abordagens de teste de segurança

Cada organização tem preocupações exclusivas de negócios e de missão, que por sua vez requerem estratégias e abordagens de teste de segurança exclusivas para identificar e mitigar os riscos de segurança. No entanto, há também algumas preocupações de segurança que são comuns em muitas organizações.

Uma abordagem de teste de segurança é definida no nível do projeto e deve ser consistente com a política e estratégia de teste da organização. A abordagem de teste de segurança de um projeto será uma combinação única de técnicas, ferramentas e habilidades para abordar os objetivos de teste de segurança para esse projeto.

Ao analisar uma situação com a finalidade de definir uma abordagem de teste de segurança, considere o seguinte:

- A origem dos sistemas ou aplicações.
- Qualquer teste de segurança anterior.
- A política de segurança.
- A política de testes de segurança
- Quaisquer avaliações de risco de segurança já realizadas na organização.
- O ambiente técnico em uso (por exemplo, tipo de software e versão, frameworks, linguagens de programação, sistemas operacionais).
- Testes de segurança na equipe de teste.
- Riscos comuns de segurança.
- A estrutura organizacional do teste.
- A estrutura da equipe do projeto.
- A experiência da equipe de teste com várias ferramentas de teste de segurança.
- Restrições (por exemplo, recursos limitados, tempo limitado, falta de acesso a ambientes).
- Suposições (por exemplo, suposições sobre outras formas anteriores de testes de segurança realizados).

Diferentes ambientes técnicos e tipos de aplicativos (por exemplo, cliente/servidor, web, mainframe) geralmente requerem estratégias de testes de segurança diferentes. Por exemplo, o desenvolvimento de um software pode exigir revisões de código para detectar vulnerabilidades de segurança no código, enquanto o teste de software pode exigir mascaramento dos dados de teste. Aplicativos baseados na Web têm vulnerabilidades diferentes dos sistemas mainframe e, portanto, exigem diferentes tipos de testes de segurança.

Algumas vulnerabilidades são comuns a várias tecnologias. Por exemplo, as vulnerabilidades de estouro de buffer podem ocorrer em aplicativos cliente-servidor, web e móveis com diferenças baseadas no gerenciamento de memória em cada tecnologia. O resultado é o mesmo em todos os ambientes, o que é um comportamento imprevisível do software que pode permitir que um invasor acesse uma aplicação e execute tarefas que normalmente não seriam permitidas.

A proteção inadequada de dados pode ocorrer em qualquer tecnologia ou ambiente. No entanto, criptografar dados nos ambientes da Web e móveis é diferente do que no ambiente mainframe. Os algoritmos de criptografia podem ser os mesmos (ou similares), mas a diferença é que os dados devem ser protegidos em trânsito pela Internet no caso de aplicações web e móveis. Em todas as tecnologias, os dados importantes devem ser armazenados em um formato criptografado. Houve incidentes em que dados importantes do mainframe foram enviados fisicamente (usando fita) para outra parte em um formato não criptografado. "*O Cattles Group, especializado em empréstimos pessoais e recuperação de dívidas, admitiu ter perdido duas fitas de backup contendo informações para cerca de 1,4 milhão de clientes*". [ComputerWeekly]

2.6.2 Análise de falhas em abordagens de teste de segurança

É necessário entender que há graus de fracasso. Só porque uma vulnerabilidade de segurança não é detectada e resolvida, isso não significa necessariamente que a abordagem de testes de segurança falhou. Existem muitas vulnerabilidades de segurança possíveis, com novas descobertas diárias. No entanto, há outros casos em que as abordagens dos testes de segurança foram inadequadas para identificar eficazmente os riscos de segurança, o que levou a que dados sigilosos e outros ativos digitais fossem comprometidos.

A análise de causa raiz pode ajudar a identificar por que uma abordagem de teste de segurança pode ter falhado. Possíveis causas incluem:

- Falta de liderança executiva no estabelecimento dos testes de segurança.
- Falta de provisão de recursos necessários para implementar a estratégia de testes de segurança (como falta de financiamento, falta de tempo, falta de recursos).

- Falta de implementação efetiva da abordagem de testes de segurança (como a falta de habilidades necessárias para executar as tarefas requeridas).
- Falta de compreensão organizacional e apoio à abordagem de testes de segurança.
- Falta de entendimento das partes interessadas e apoio à abordagem de testes de segurança.
- Falta de compreensão dos riscos de segurança.
- Falta de alinhamento entre a abordagem de teste e a política de segurança da organização
- Falta de alinhamento entre a abordagem de teste e a política de estratégia de testes de segurança da organização.
- Falta de compreensão da finalidade do sistema.
- Falta de informações técnicas sobre o sistema (causando suposições erradas).
- Falta de ferramentas eficazes para testes de segurança.
- Falta de habilidades em testes de segurança.

2.6.3 Identificação das partes interessadas

Para que um esforço de teste de segurança seja eficaz, um plano de negócio para ele deve ser feito à gerência. Este caso de negócio deve definir claramente os riscos de falhas de segurança e os benefícios de se ter uma abordagem de teste de segurança eficaz para um determinado projeto.

As diferentes partes interessadas verão diferentes benefícios em uma abordagem de testes de segurança:

- A gerência executiva verá a proteção do negócio como um benefício.
- A alta administração pode ver a devida diligência.
- Os clientes podem ver a proteção contra fraude.
- Os agentes de conformidade (para políticas internas de segurança corporativa) podem ver garantias de que a organização está em conformidade com as obrigações legais.
- Os reguladores (para leis de segurança externa) podem ver um benefício que as regulamentações de segurança estão sendo seguidas.
- Os agentes de privacidade podem ver o benefício que os dados privados são mantidos seguros e a devida diligência tem sido demonstrada na proteção de ativos digitais.

2.7 Melhorar as práticas de testes de segurança

Para melhorar as práticas de testes de segurança, primeiramente é necessária uma avaliação das práticas existentes. Deve haver uma maneira objetiva de realizar a avaliação. Estes são baseados em métricas-chave para os objetivos dos testes de segurança, a partir dos quais é possível identificar o grau de sucesso dos elementos-chave da estratégia.

Estas práticas devem ser avaliadas da seguinte forma:

- A partir de uma perspectiva de curto e longo prazo.
- Considerar processo e organização.
- Considerar pessoas, ferramentas, sistemas e técnicas.

As principais métricas incluem, mas não estão limitadas a:

- Níveis de cobertura de riscos de segurança por testes.
- Níveis de cobertura de políticas e práticas de segurança por testes.
- Níveis de cobertura dos requisitos de segurança por testes.
- Níveis de eficácia dos esforços anteriores de testes de segurança, com base em quando e onde as vulnerabilidades de segurança foram identificadas. Isso inclui vulnerabilidades de segurança antes e após um *release*.

3 Processos de teste de segurança (140 min)

Palavras-chave

Coleta de contas, quebra de senha, engenharia social, abordagem de teste, plano de teste, processo de teste.

Objetivos de Aprendizagem

3.1 Definição do processo de teste de segurança

AS-3.1.1 (K3): Para um determinado projeto, demonstrar a capacidade de definir os elementos de um processo de teste de segurança eficaz.

3.2 Planejamento do teste de segurança

AS-3.2.1 (K4): Analisar um determinado plano de teste de segurança, dando *feedback* sobre os pontos fortes e fracos do plano.

3.3 Projeto de teste de segurança

AS-3.3.1 (K3): Para um dado projeto, implementar testes de segurança conceituais (abstratos), com base numa determinada abordagem de teste de segurança, juntamente com riscos de segurança funcionais e estruturais identificados.

AS-3.3.2 (K3): Implementar casos de teste para validar políticas e procedimentos de segurança.

3.4 Execução do teste de segurança

AS-3.4.1 (K2): Compreender os elementos-chave e as características de um ambiente de teste de segurança eficaz.

AS-3.4.2 (K2): Compreender a importância de planejar e obter aprovações antes de realizar qualquer teste de segurança.

3.5 Avaliação do teste de segurança

AS-3.5.1 (K4): Analisar os resultados dos testes de segurança para determinar:

- A natureza da vulnerabilidade da segurança.
- A extensão da vulnerabilidade de segurança.
- O potencial impacto da vulnerabilidade de segurança.
- As sugestões de remediação.
- Os métodos de relatório de teste.

3.6 Manutenção do teste de segurança

AS-3.6.1 (K2): Compreender a importância de manter os processos de testes de segurança, dada a natureza evolutiva da tecnologia e das ameaças.

3.1 Definição do processo de teste de segurança

Como os testes de software em geral, os testes de segurança também são uma atividade do ciclo de vida. A falha em implementar e testar defesas de segurança em um projeto pode levar a graves defeitos de segurança que podem nunca serem completamente resolvidos. O processo de teste de segurança deve ser alinhado com o processo de desenvolvimento para que as atividades de teste apropriadas sejam realizadas quando necessário.

Os riscos e as necessidades dos testes de segurança de cada organização serão únicos devido à natureza da organização, aos ambientes técnicos, ao processo de desenvolvimento de software e aos riscos de negócio. Portanto, o processo de teste de segurança deve ser definido no contexto desses fatores.

3.1.1 Processo de teste de segurança ISTQB

A Tabela 3.1, mostra a relação entre o processo geral de teste do ISTQB, conforme descrito nos programas ISTQB *Foundation Level* e *Advanced Level*, e o ISTQB *Security Test*. Exemplos de tarefas de teste de segurança são mostradas para cada etapa do processo.

3.1.1.1 Tabela 3.1 - Processo de Teste de Segurança ISTQB

Processo de Teste ISTQB	Processo ISTQB Security Test	Exemplo Security Test
Planejamento e Controle de Teste	*Planejamento e Controle de Testes de Segurança* O objetivo é definir um escopo apropriado de testes que corresponda aos riscos de segurança.	• Definir objetivos de teste de segurança • Definir o escopo dos testes de segurança • Identificar os recursos de teste de segurança • Definir estimativas e cronogramas de teste de segurança • Definir métricas de teste de segurança, critérios de entrada e saída • Monitorar o progresso e os resultados dos testes de segurança • Tomar as medidas necessárias em resposta às informações aprendidas durante outras atividades de teste de segurança.
Análise e Modelagem de Teste	*Análise e Modelagem de Testes de Segurança* O objetivo é obter compreensão de ameaças e riscos de segurança específicos com base em avaliações de segurança, auditorias e fontes padrão de vulnerabilidades conhecidas.	• Revisar itens que servem como base de testes de segurança, como avaliações de riscos de segurança, requisitos de segurança e políticas de segurança • Definir condições de teste de segurança com base em: • Objetivos de teste • Riscos de segurança • Padrões de segurança e vulnerabilidades conhecidas • Defesas implementadas para proteger o sistema e seus dados • Escopo dos testes de segurança • Aplicabilidade de ferramentas de teste de segurança

Processo de Teste ISTQB	Processo ISTQB Security Test	Exemplo Security Test
Implementação e Execução do Teste	*Implementação e Execução de Testes de Segurança* O objetivo é traduzir testes conceituais em testes que podem ser executados manualmente ou com ferramentas. Além disso, o objetivo é realizar esses testes usando uma variedade de perspectivas de teste de segurança - usuário interno, usuário externo, usuário mal-intencionado, etc.	• Criar casos de teste de segurança, cenários de teste, scripts de teste ou outras especificações de teste. • Realizar testes de segurança funcionais com base em especificações de segurança definidas. • Realizar testes de segurança funcional e penetração. • Sobre o conhecimento e intuição do testador. • Realizar testes de segurança com base em um modelo de um sistema. • Configurar ou preparar um ambiente de teste para realizar testes de segurança.
Avaliação dos Critérios de Saída e Relatórios	*Avaliação e Relatórios Resultados do Teste de Segurança* Isso é frequentemente realizado juntamente com a execução do teste para avaliar testes individuais e para relatar novas ameaças o mais rápido possível.	• Determinar vulnerabilidades de segurança específicas com base nos resultados dos testes. • Avaliar os níveis de risco de segurança com base nos resultados dos testes de segurança realizados. • Relatar os resultados dos testes de segurança intermédios e finais. • Gestão e outras partes autorizadas.
Encerramento do Teste	*Encerramento do teste* O objetivo é colocar as atividades de teste de segurança em um ponto de encerramento para que os testes possam ser mantidos e executados regularmente para suportar novos requisitos de segurança e / ou detectar novas ameaças. Além disso, todos os testes e resultados de segurança são armazenados de forma segura, estando disponíveis para uso se necessário em testes de segurança futuros.	• Certificar de que todos os testes de segurança planejados foram realizados. • Determinar se os resultados de testes de segurança (relatórios) foram entregues. • Arquivar resultados de teste, dados de teste e outras informações confidenciais em locais seguros. • Analisar os resultados dos testes de segurança para melhorar o desenvolvimento de sistemas e aplicativos em termos de segurança.

Tabela 3.1

É importante entender que o *ISTQB Security Tester* não é necessariamente de natureza sequencial. O processo de teste de segurança deve ser alinhado com o processo de ciclo de vida do software da organização. Uma importante implicação do processo descrito nesta seção é que as atividades de teste de segurança são realizadas em paralelo a outras atividades e testes do ciclo de vida do projeto.

Além disso, as tarefas de teste de segurança mostradas na Tabela 3.1 servem como exemplos e não como requisitos prescritivos para tarefas de teste de segurança. As tarefas de teste de segurança exatas para uma

organização dependem da estratégia de teste de segurança e da abordagem adotada pela organização, conforme mostrado na figura 3.1 a seguir.

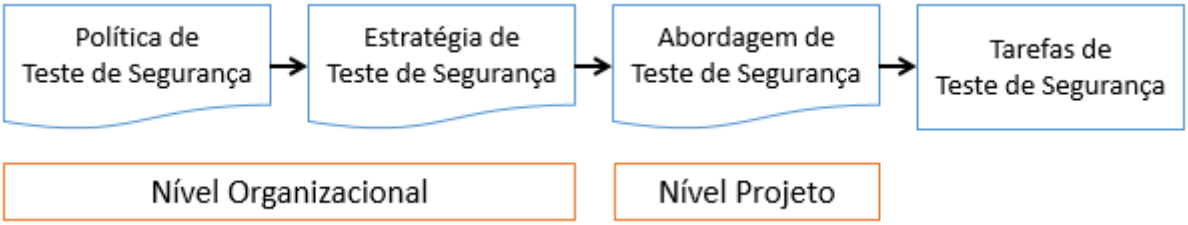

Figura 3.1 - Hierarquia de Planejamento de Testes de Segurança

3.1.2 Alinhando o processo de teste de segurança a um modelo de ciclo de vida de software

Cada um dos seguintes tipos de processos de ciclo de vida tem preocupações de teste de segurança. É importante alinhar os testes de segurança de acordo com o ciclo de vida.

3.1.2.1 Ciclos de vida sequenciais

Nesses projetos, o testador em segurança deve estar ciente do seguinte:

- As necessidades e riscos de segurança são definidas no início de um projeto e devem ser documentadas nas especificações de requisitos de software.
- As necessidades de segurança podem mudar durante o projeto, mas podem não se refletir em requisitos de software atualizados. Os testes de segurança podem parecer muito específicos e completos, mas podem não estar completos ou atuais devido a riscos tardios.
- Testes de segurança podem ser realizados a qualquer momento, mas é comum que esses testes sejam realizados no final do projeto.
- Pode ser difícil abordar os resultados dos testes de segurança no final de um projeto de ciclo de vida sequencial.

3.1.2.2 Ciclos de vida iterativos / incrementais

Projetos incrementais fornecem lançamentos pequenos e frequentes versões de um aplicativo. Métodos ágeis são um exemplo desta abordagem. Nesses projetos, o testador em segurança deve estar ciente do seguinte:

- As necessidades e os riscos de segurança surgem durante todo o projeto (normalmente no contexto de uma iteração ou *sprint*) e podem ser definidos em especificações de requisitos, estórias de usuários, modelos, critérios de aceitação e protótipos.
- As necessidades e riscos de segurança podem mudar durante o projeto e podem (devem) ser abordados na iteração em que são identificados.
- Testes de segurança podem ser realizados continuamente ao longo de um projeto.
- Dependendo da natureza do risco de segurança, pode não ser possível mitigar e testar completamente durante um curto ciclo de liberação.

3.1.2.3 Software comercial de prateleira (COTS)

Estes projetos são muitas vezes caixa-preta na natureza e podem ou não ser personalizados. Eles geralmente contêm vulnerabilidades de segurança na medida em que frequentes atualizações de segurança e patches são necessários. Não há acesso ao código, portanto a análise estrutural e os testes estruturais não são possíveis.

3.1.2.4 Software de código aberto

Esta é uma variante do COTS, mas com uma distinção importante - o código está disponível para todo o mundo. Esses produtos também têm vulnerabilidades de segurança, portanto, é de vital importância que os patches de segurança sejam mantidos atualizados. Uma vez que uma vulnerabilidade de segurança tenha sido divulgada, os usuários dessa versão específica do software (e anteriores) correm o risco de sofrer um ataque.

3.1.2.5 Exemplo: Processo de teste de segurança em um ciclo de vida sequencial

É importante notar que os testes de segurança não precisam ser confinados a uma fase ou atividade em um projeto. É particularmente importante evitar a situação em que os testes de segurança (e outros testes) não são realizados até à fase de aceitação do projeto. No final do projeto, é especialmente oneroso e arriscado lidar com quaisquer defeitos descobertos. As tarefas de teste de segurança apropriadas que devem ser realizadas em cada fase do ciclo de vida sequencial são:

- **Requisitos**: Os requisitos de segurança são definidos e revistos como parte do esforço de requisitos gerais para expressar as necessidades da organização. É também onde os casos de uso podem ser escritos. É neste momento que uma abordagem de teste de segurança deve ser desenvolvida.
- **Análise e projeto:** Normalmente, alguém no papel de analista de negócios examinará a declaração inicial de requisitos e refiná-los-á para preencher lacunas. Um analista de sistemas ou arquiteto, em seguida, analisará os requisitos para propor a maneira ideal para entregar uma solução que atenda às necessidades do usuário. Neste caso, a segurança seria uma das necessidades funcionais e não-funcionais, juntamente com outras como usabilidade e eficiência. Neste ponto, os criadores de testes de segurança podem ter uma ideia da arquitetura e do que precisa ser testado, tanto a partir das perspectivas de segurança estrutural e funcional. Principais objetivos de teste de segurança devem ser definidos neste momento.
- **Modelagem detalhada**: Neste ponto, as interfaces de usuário e bancos de dados estão projetadas. As regras funcionais são refinadas e o projeto de teste de segurança torna-se mais detalhado. Os primeiros testes de segurança podem ser realizados com base em modelos.
- **Codificação e implementação**: Isto é quando as especificações de projeto são implementadas como código. Essa é a primeira oportunidade de testar a estrutura do aplicativo, incluindo testes de vulnerabilidades de segurança, como defeitos de estouro de buffer e edições de campo que podem permitir que a *SQL injection* ocorra. A análise estática e as revisões de código são muito valiosas nesta fase e devem incluir o exame do código sob a perspectiva da segurança. Teste de componentes também é uma atividade chave para verificar se o código funciona conforme especificado. Os testes de integração entre componentes também podem começar como componentes que interagem uns com os outros tornam-se disponíveis para testes em montagens pequenas.
- **Teste do sistema**: Este é o teste de sistemas e subsistemas. O teste do sistema inclui software, hardware, dados, procedimentos e como as pessoas interagem com o sistema. Esses testes são muitas vezes de natureza transacional para testar os processos de negócios. As bases para o teste do sistema podem ser modelos de modelagem, requisitos, casos de uso e quaisquer outras especificações que transmitam a perspectiva do sistema. Além disso, os testes de integração de sistemas podem precisar ser realizados para testar como vários sistemas se comunicam e trocam dados. Testes de segurança nesta fase assume uma visão mais ampla porque o hardware e os intercâmbios de dados estão envolvidos. A segurança da transação pode ser testada, o que inclui autenticação, armazenamento de dados, implementação de firewall, bem como controles de segurança processuais.
- **Testes de Aceite de Usuário**: Isto acontece quando o teste confirma que um sistema suporta processos de negócios do mundo real e pode abranger vários sistemas em várias organizações. O objetivo dessa fase não é tanto encontrar defeitos, mas sim validar que o sistema atenda às necessidades dos usuários em condições reais. Isso inclui garantir que os requisitos de segurança foram implementados e atendidos

corretamente. Nesta fase, os testes de segurança já deveriam ter sido realizados em grande parte, mas ainda há oportunidades para testar cenários de segurança que ocorrem no nível do processo de negócios.

- **Implantação**: É quando o sistema concluído e testado é implantado para os usuários. Há muitas maneiras em que isso pode ocorrer, como nas implantações piloto para grupos selecionados ou uma implantação maciça para todos os usuários. Outra abordagem é uma implantação paralela onde um sistema antigo e um novo sistema estão em operação simultânea por um tempo limitado. Grande parte da decisão de uma implementação de corte direto depende do risco de implantação para todos os usuários, e da confiança adquirida durante os testes de aceitação. A segurança é uma preocupação durante a implantação do sistema, já que todos os componentes do sistema devem ser implantados de forma que nenhuma nova vulnerabilidade seja introduzida. Isso pode ocorrer se as configurações de segurança não estiverem corretas no ambiente de destino. Um exemplo disso seria se os direitos de acesso ao banco de dados não estiverem corretos no ambiente de produção.
- **Manutenção**: A medida que novas necessidades surgem ou os defeitos são descobertos após a implantação, a manutenção é executada. O teste assume uma dimensão diferente, pois o foco está em testar as alterações e realizar testes de regressão. Testes de segurança também devem ser realizados para garantir que novas vulnerabilidades não sejam introduzidas durante as alterações. Parte do processo de manutenção é manter os firewalls e outras tecnologias de segurança atuais. O monitoramento contínuo do sistema pode detectar atividades suspeitas que podem precisar ser tratadas imediatamente.

3.1.2.6 Exemplo: Processo de teste de segurança em um ciclo de vida iterativo / incremental

Há uma variedade de metodologias que foram introduzidas nos últimos 20 anos para definir software de construção em incrementos ou iterações menores. Neste exemplo, lançamentos do software são entregues a cada quatro semanas. A base de trabalho (e de teste) são as estórias de usuários, cada um com critérios de aceitação definidos.

A seleção de recursos para construir e entregar baseia-se em um *backlog* priorizado. Os recursos selecionados devem refletir os itens que oferecem o maior valor e são alcançáveis no período de *sprint*. O testador em segurança trabalha com o negócio e/ou proprietário do produto para ter os requisitos de segurança adequados e corretos.

Neste exemplo, quatro principais funções de segurança são selecionadas para a primeira iteração porque elas serão necessárias para desenvolver muitos dos outros recursos. Os recursos são:

- *Login* de usuário.
- Ativação de SSL (*Secure Socket Layer*).
- Redefinição de senha perdida.
- Bloqueio de conta após três tentativas malsucedidas

Cada uma dessas características é escrita como estórias de usuário e refinadas em requisitos mais detalhados, cada um com critérios de aceitação.

Do ponto de vista do teste de segurança, o testador em segurança trabalha com o desenvolvedor para garantir que as políticas e os protocolos corretos sejam refletidos no código. O testador em segurança também trabalhará ao lado do desenvolvedor para testar as funções à medida que forem sendo desenvolvidas.

Neste exemplo, a primeira versão pode ser apenas a página de *login* e funções associadas para o *login*, como redefinir uma senha perdida e o controle de bloqueio. Na próxima iteração, serão desenvolvidas outras funções, com base na prioridade para as partes interessadas. Em cada iteração, o testador em segurança testará para certificar-se de que os controles de segurança estão funcionando corretamente e nenhuma nova vulnerabilidade de segurança foi introduzida. As iterações continuarão até que todas as tarefas de *backlog* tenham sido concluídas.

Em ambos os exemplos (iterativo/incremental e sequencial), as etapas do processo de teste de segurança podem ser vistas como tarefas integrais para garantir uma aplicação segura.

3.2 Planejamento de teste de segurança

3.2.1 Objetivos de planeamento de teste de segurança

Os testes de segurança em geral devem se concentrar em dois aspectos:

- Verificar se as defesas de segurança projetadas são implementadas e funcionam conforme o planejado.
- Verificar se não há vulnerabilidades introduzidas durante o desenvolvimento do aplicativo.

Como mencionado anteriormente neste *syllabus*, todas as defesas de segurança a serem implementadas devem ser baseadas em uma análise de risco. Isso fornece um ponto de partida ao planejar testes de segurança para um projeto.

Muitas das vulnerabilidades desenvolvidas não intencionais podem ser evitadas usando atividades de garantia de qualidade e melhores práticas durante as atividades de arquitetura, modelagem e codificação. Testar se as vulnerabilidades são introduzidas começa com uma avaliação das práticas utilizadas pela equipe de desenvolvimento. Com base no resultado, pode ser necessário selecionar e introduzir testes de segurança adicionais.

3.2.2 Elementos chave do plano de teste de segurança

Os principais elementos de um plano de teste de segurança estão listados abaixo. Cada um destes pode ser determinado por fazer as perguntas específicas para um determinado projeto.

- Identificar o escopo dos testes de segurança.
  - O que está dentro ou fora do escopo?
  - O que é alcançável sobre os recursos do projeto, os riscos de segurança e as restrições de tempo?

- Identificar quem deve realizar os testes de segurança.
  - A organização tem pessoas com habilidades de teste de segurança apropriadas?
  - A organização se sente confortável com testes de segurança de terceirização?
  - No caso de software comercial e software desenvolvido pelo fornecedor, que testes de segurança são da responsabilidade do fornecedor e quais são da responsabilidade do cliente?
  - Os testadores em segurança precisam de treinamento no uso de ferramentas de teste de segurança específicas?

- Atribuir o cronograma apropriado para os testes de segurança, tendo em conta outros requisitos de programação de teste do projeto.
  - Que itens relacionados à segurança precisam ser implementados e testados antes que outros testes ocorram? (Por exemplo, direitos de acesso e *logins*)
  - Quando os recursos de segurança estarão disponíveis para testes?
  - Quanto tempo levará para realizar testes de segurança dados os recursos planejados e o escopo?

- Definir as tarefas a serem executadas e o tempo necessário para cada.
  - Quanto tempo é necessário para projetar os testes de segurança apropriados com base nos recursos planejados e no escopo?
  - Quanto tempo é necessário para avaliar e relatar os resultados dos testes de segurança?

  - Quanto tempo é necessário para realizar testes de regressão relacionados à segurança?
  - Quanto tempo é necessário para estabelecer o ambiente de teste de segurança?

- Definir o(s) ambiente(s) de teste de segurança.
  - Qual é a extensão do ambiente? (Plataforma, tecnologia, tamanho, localização)
  - Este é um novo ambiente?
  - Que ferramentas de teste de segurança e outras ferramentas de teste precisam ser instaladas no ambiente?

- Obtenção de autorizações e aprovações para atividades de teste de segurança.
  - Quem deve autorizar e aprovar os testes de segurança?
  - Quando essa autorização é necessária?
  - O orçamento e o financiamento são suficientes?

Como qualquer projeto entregue, o plano de teste de segurança deve ser revisto para avaliar a integridade e a exatidão. Uma vez que os testes de segurança são muitas vezes de natureza técnica, uma sessão de revisão técnica pode ser o método mais adequado. No entanto, acompanhamentos e inspeções também podem ser utilizados.

Uma lista de verificação padrão pode ajudar a formar a base do que é coberto em uma sessão de revisão. Como qualquer outra revisão, o feedback deve ser construtivo e não destinado ao produtor do plano de teste de segurança. A equipe de revisão deve incluir pessoas conhecedoras de todas as áreas afetadas pelos aspectos de segurança discutidos no plano de testes de segurança.

Os membros da equipe de revisão podem não ser necessariamente testadores em segurança ou possuírem experiência em segurança. Por exemplo, o gerente de uma unidade de negócios pode ter informações sobre riscos de segurança que devem ser registradas no plano de teste de segurança. Os auditores de TI e os administradores de segurança são especialmente úteis nas revisões dos planos de teste de segurança devido ao seu conhecimento das políticas e procedimentos de segurança.

3.3 Projeto de teste de segurança

Há várias maneiras de iniciar o projeto de teste de segurança. Por exemplo, pode ser iniciado baseado em:

- Uma análise de risco realizada.
- Um modelo de ameaça disponível.
- Uma classificação de origem *ad hoc* de riscos de segurança (ver [BSTQB_ATTA_SYL]).

Qualquer um destes pode formar uma base viável para um projeto de teste de segurança.

Dependendo do tipo de projeto, é importante garantir que há testes de segurança em cada fase de desenvolvimento aplicável.

3.3.1 Modelagem de teste de segurança

Testes de segurança detalhados são baseados nos riscos de segurança, uma estratégia de teste de segurança e outras fontes, como modelos de ameaças. Testes de segurança também podem ser vistos como funcional e estrutural na natureza. Por exemplo, no caso do teste de segurança de um site de comércio eletrônico, os riscos de segurança funcionais podem ser *SQL injection*, coleta de contas e quebra de senhas. Um exemplo de um risco de segurança estrutural seria uma condição de estouro de buffer que permitiria a um invasor obter acesso através de uma falha de memória.

Os seguintes são atributos essenciais de testes detalhados de segurança:

- Priorizada por riscos de segurança e modelos de ameaça identificados.
- Rastreado para requisitos de segurança definidos.
- Definido com base no público-alvo (desenvolvedores, testadores funcionais e de segurança).
- Definido com base em perfis de defeitos de segurança.
- Se aplicável, ser projetado para ser automatizado.

O fluxo básico do projeto de teste de segurança pode ser visto como:

1. A abordagem dos testes de segurança (nível do projeto).
2. Riscos de teste de segurança, modelos de ameaças e requisitos (nível de projeto).
3. Técnicas de projeto de teste de segurança (com base em riscos, requisitos e aplicação).
4. Casos e cenários de teste de segurança.

No restante deste capítulo, riscos e vulnerabilidades de segurança comuns são apresentados juntamente com a técnica de modelagem de teste de segurança associada. Novos riscos e vulnerabilidades de segurança surgem rapidamente, por isso é aconselhável que os planejadores de teste de segurança permaneçam atualizados com os padrões de segurança e as listas de ameaças, conforme mencionado no Capítulo 9.

Um princípio fundamental é que um processo de modelagem de teste de segurança deve ser capaz de criar e implementar testes com base em qualquer risco, exigência ou ameaça de segurança identificados.

3.3.1.1 Controles de segurança funcionais (por exemplo, controles de transação)

Esses testes são projetados para verificar e validar que os controles funcionam corretamente e são eficazes na detecção e prevenção de ações não autorizadas.

Exemplo: Um caixa de banco não pode autorizar uma retirada em dinheiro de uma certa quantia sem a aprovação do chefe de caixa aderida ao sistema.

3.3.1.2 Controles de acesso funcionais (por exemplo, *logins*, senhas, *tokens*)

Esses testes são talvez o que a maioria das pessoas pensam imediatamente em termos de testes de segurança. Os testes incluem:

- Políticas de nome de usuário e senha são aplicadas corretamente.
- Se o nível de controle de acesso é adequado para o risco.
- Se os controles de acesso são resistentes a programas de quebra de senhas.

Exemplo: A coleta de contas é a prática de identificar um nome de usuário. Uma vez que o nome de usuário é adivinhado ou identificado, a senha é a peça restante necessária para obter acesso ao sistema. Um teste comum é verificar que quando um nome de usuário correto é inserido com uma senha incorreta, a mensagem de erro não indique qual dos itens está incorreto.

3.3.1.3 Controles de acessos estruturais (p.e., direitos de acesso de usuários, criptografia, autenticação)

Os testes para esses controles são baseados em como os direitos do usuário foram estabelecidos para acesso aos dados, funcionalidades e níveis de privacidade. Os controles de acesso estrutural são normalmente aplicados por um administrador de sistemas, ou de segurança, ou de banco de dados. Em alguns casos, os direitos de acesso são uma opção de configuração em um aplicativo. Em outros casos, os direitos de acesso são aplicados no âmbito da infraestrutura do sistema.

Os testes de controles de acesso estruturais incluem a criação de contas de usuário de teste para cada nível de acesso de segurança e a verificação de que cada nível de acesso não tem direitos de acesso restritos para esse nível. Por exemplo, as contas de usuário seriam criadas para um nível mínimo de acesso. Testes devem

ser realizados para garantir que um usuário com acesso mínimo não possa executar atividades de acesso no nível de gerente ou administrador.

3.3.1.4 Práticas de codificação segura

Este é basicamente um método de teste estático para determinar se os desenvolvedores de software e sistema estão seguindo os métodos de segurança estabelecidos à medida que eles criam aplicativos.

Um princípio-chave é que muitos ataques de segurança são realizados através da exploração de defeitos de software para causar um sistema para se comportar de forma inesperada.

Uma lista muito curta de práticas de codificação segura inclui:

- Se algoritmos e controles de gerenciamento de sessão comprovados são usados para criar identificadores de sessão aleatórios.
- Se as decisões de autorização são feitas apenas por objetos confiáveis do sistema sob controle da organização que fornece autorização (por exemplo, a autorização deve ocorrer no lado do servidor).
- Se as informações de segurança não sejam apresentadas em mensagens de erro. Essas informações podem incluir detalhes do sistema, identificadores de sessão e informações de conta.
- Se os erros de aplicativo são manipulados dentro do aplicativo, em vez de confiar na configuração do servidor.
- Se as solicitações HTTP GET não incluam informações confidenciais.

Os manipuladores de erro não devem exibir rastreamento de pilha ou outras informações de depuração.

- Todas as falhas de validação de entrada de dados devem ser registradas.
- Qualquer informação sigilosa que possa ser armazenada temporariamente no servidor deve ser protegida (por exemplo, a criptografia deve ser usada). Esta informação segura temporária deve ser removida quando não for mais necessária.
- Um aplicativo não deve ser capaz de emitir comandos diretamente para o sistema operacional. Em vez disso, as APIs internas devem ser usadas para realizar tarefas do sistema operacional.
- As senhas, sequências de conexão ou outras informações confidenciais não devem ser armazenadas em texto não criptografado nas máquinas dos clientes (por exemplo, em *cookies*). Devem ser proibidas as incorporações de tais informações em formatos não seguros, como Adobe Flash, código compilado e o MS Viewstate.
- A criptografia deve ser usada para a transmissão de todas as informações confidenciais. O *Transport Layer Security* (TLS) é uma maneira de proteger os dados em trânsito ao usar conexões HTTP. Para conexões não-HTTP, a criptografia deve ser usada para transmitir informações confidenciais.
- Os dados fornecidos pelo usuário não devem ser passados diretamente para qualquer função dinâmica "*include*".
- Todos os dados fornecidos pelo usuário devem ser adequadamente sanitizados e validados antes de serem usados pelo aplicativo.
- As variáveis devem ser fortemente digitadas em idiomas que suportem a verificação de tipos. Ou seja, as variáveis devem ter um tipo de entrada definida. Por exemplo, um campo numérico não deve aceitar caracteres alfa. Esta restrição seria definida na definição de tipo da variável, bem como na base de dados. É possível escrever código seguro em *JavaScript* ou em outros idiomas que não suportam a verificação de tipo cumprida.
- Em vez de usar o novo código não gerenciado para tarefas comuns, use o código testado, confiável e aprovado que esteja sob o gerenciamento de configuração.
- Executar serviços com o mínimo de privilégios possíveis (nunca em root) e cada serviço deve ter sua própria conta de usuário no sistema operacional.

Uma lista de práticas de codificação segura pode ser encontrada no Guia de Referência Rápida *OWASP Secure Coding Practices* [OWASP1] e *Top 10 Secure Coding Practices* [CERT1]. Além disso, o SANS compila uma lista dos 25 erros de software mais perigosos em [SANS1].

Testes dinâmicos podem ser realizados para determinar se práticas como validação de dados e mensagens de erro foram seguidas por desenvolvedores. Além disso, uma das vulnerabilidades de segurança mais comuns, o estouro de buffer de memória, pode ser identificado com ferramentas de teste de memória dinâmica.

3.3.1.5 Acesso ao sistema operacional

Uma vez que o acesso é adquirido ao sistema operacional, um invasor pode controlar dados, acessar a rede e implantar *malware*. Os testes para isso podem incluir testes para a capacidade de plantar *rootkits* e outros códigos maliciosos em um sistema.

3.3.1.6 Vulnerabilidades de linguagem (por exemplo, Java)

De acordo com pesquisadores de segurança da *WhiteHat Security*, um fornecedor de segurança de aplicativos, não houve diferenças significativas entre linguagens quando se trata de vulnerabilidades de segurança. [WhiteHat Security, 2014]. Em abril de 2014, a *WhiteHat Security* publicou um Relatório de Estatísticas de Segurança de Website baseado em avaliações de vulnerabilidade realizadas contra 30.000 sites de clientes usando um scanner proprietário e os resultados indicaram diferenças insignificantes na segurança relativa de idiomas como .NET, Java , PHP, ASP, ColdFusion e Perl. Essas seis línguas compartilhavam um número médio relativamente pequeno de vulnerabilidades, e problemas como *SQL injection* e a vulnerabilidades de scripts entre sites permaneciam difusos. [WhiteHat Security, 2014]. É importante reconhecer que o código seguro pode ser alcançado com muitas linguagens, assim como o código não seguro. O fator chave é como um aplicativo é codificado (implementado) em qualquer linguagem.

A Divisão CERT do Instituto de Engenharia de Software fornece publicações [CERT2] e ferramentas [CERT3] que abordam questões de segurança específicas de linguagem. Além disso, o *Vulnerability Notes Database* [CERT4] fornece informações oportunas sobre vulnerabilidades de softwares. As notas de vulnerabilidade incluem resumos, detalhes técnicos, informações de correção e listas de fornecedores afetados.

3.3.1.7 Vulnerabilidades de plataforma (por exemplo, Windows, Linux, Mac OS, iOS, Android)

Cada plataforma de computação tem seu próprio conjunto de vulnerabilidades de segurança. A preocupação com o testador em segurança é garantir que as atualizações de segurança da plataforma sejam aplicadas prontamente e em todos os dispositivos que executam a plataforma afetada.

3.3.1.8 Ameaças externas

As ameaças externas à segurança são aquelas que a maioria das pessoas conhece quando se trata de ataques cibernéticos. Algumas ameaças externas, como a exploração de vulnerabilidades de aplicativos ou de linguagem, podem ser detectadas, testadas e prevenidas.

Um Ataque de Negação de Serviço (DoS) é outro tipo de ameaça externa. Em geral, esses ataques baseiam-se na sobrecarga dos recursos do sistema ou aplicativo, de tal forma que o sistema ou aplicativo se torna inacessível para usuários legítimos. Os ataques *DoS* podem ser direcionados à largura de banda da rede, sistema ou conectividade de aplicativo, ou serviços, ou funções específicas.

Um Ataque Distribuído de Negação de Serviço (DDoS) é um tipo de ataque *DoS* onde o ataque é iniciado indiretamente usando outros recursos do computador. Possíveis técnicas são a amplificação ou o uso de *botnets*, que são muitos computadores comprometidos anteriormente sob controle ou comando de um atacante. Um atacante pode ganhar controle simplesmente criando infecções de vírus ou usando *Trojans*. Os computadores infectados podem ser usados como agentes, cada um enviando tráfego para uma vítima específica (rede) conforme o alvo do atacante.

Ao usar ataques de amplificação ou reflexão, o atacante está usando uma vulnerabilidade (ou mesmo a funcionalidade desejada) em protocolos específicos (por exemplo, DNS ou NTP). O atacante envia uma grande quantidade de tráfego para endereços de *broadcast IP* (vários *hosts*) contendo o endereço de origem falsificado da vítima. Isso resulta no serviço de difusão desse tráfego para o endereço da vítima multiplicando a quantidade original de tráfego com o número de *hosts*. Quando um atacante envia estes tipos de pedidos inúmeras vezes por segundo, a vítima é confrontada com o elevado número de respostas que tem de enviar.

Exemplo: O Atacante A envia uma solicitação para o sistema B para obter uma lista completa de todos os registros DNS conhecidos enquanto personifica a Vítima C, muitas vezes com um endereço IP falsificado. O sistema B enviará a lista completa para a Vítima C inundando-a com uma quantidade ampliada de dados.

Outra forma de ataques *DoS* são ataques de exaustão de recursos. Esses tipos de ataques abusam de uma funcionalidade desejada, consumindo os recursos de computação (CPU, memória, armazenamento em disco, etc.) necessários para fornecer a funcionalidade.

Exemplo: Uma das funcionalidades do protocolo SSL é a opção de gerar novas chaves em uma sessão existente se o cliente ou servidor suspeitar de uma sessão comprometida. Gerar chaves é um processo caro. Quando um invasor envia uma solicitação para gerar novas chaves várias vezes por segundo, um sistema mal configurado ou desprotegido pode acabar em uma situação em que ele está gerando apenas novas chaves e não tem recursos para fazer outras coisas.

Por último, existem os chamados ataques *DoS* lógicos, em que um intruso pode abusar da funcionalidade pretendida para impedir que outros utilizadores acessem o sistema.

Exemplo: Um aplicativo usa nomes de usuário previsíveis e bloqueia um usuário permanentemente após três tentativas de *login* errados. Um invasor pode adivinhar os nomes de usuários e bloquear muitas contas no sistema, fazendo com que muitos usuários não possam acessá-lo (e indiretamente, fazer o serviço de *helpdesk*).

Há quatro níveis de teste para DDoS.

1. Teste para garantir que os computadores não estejam infectados com um *malware* conhecido.
2. Testar a capacidade dos sistemas de detecção de intrusão para identificar rapidamente múltiplas solicitações de um único computador em um curto período.
3. Identificar configurações que permitem funcionalidades que podem ser abusadas por um invasor (por exemplo, SSL, servidor web, DNS).
4. Identificar defeitos lógicos que podem permitir um DoS.

Intrusões são outra forma de ataque externo. Há muitas maneiras de realizar intrusão externa em um sistema. Esses ataques baseiam-se em alguém "invadindo" um sistema para obter informações. Alguns dos métodos são descritos na lista abaixo:

- Engenharia social.
- Ataques de injeção (*SQL injection*, código malicioso – *script injection*).
- Compromisso da conta (colheita, redefinição de senha).
- Explorando vulnerabilidades conhecidas (*firewall*, SO, *framework*, aplicativo).
- Ataques de *malware.*
- Iniciativas de configuração inseguras.
- Defeitos de autorização de acesso.
- Ataques de lógica de aplicativo (aproveitando os defeitos de aplicativo, especialmente em aplicativos baseados na Web, para usar indevidamente a função - por exemplo, executar etapas fora de ordem em um aplicativo de compras de comércio eletrônico para obter desconto ou crédito)

Interceptar uma transmissão de rede enviada de dentro de uma organização para alguém de outra organização não é considerado um ataque de intrusão, mas sim uma violação interna.

3.3.1.9 Ameaças internas

As maiores ameaças podem ser internas. Considere as seguintes fontes de ataque interno:

- Espionagem corporativa onde um empregado confiável pode vender informações corporativas, incluindo informações de contas de clientes, segredos comerciais, informações de acesso de funcionários, etc.
- Informações obtidas por desenvolvedores terceirizados, testadores e outros funcionários (como representantes de atendimento ao cliente). Às vezes as pessoas deixam o emprego de uma empresa de terceirização e levam as informações com eles em sua memória.
- O roubo de discos rígidos e outros dispositivos físicos de armazenamento.
- Funcionários descontentes que procuram danificar a empresa por vazamento de informações confidenciais ou cometer atos de roubo, pagando-se dinheiro sob a aparência de faturas legítimas (mas fabricadas).

3.3.1.10 Formato e estrutura do teste de segurança

Cada organização que executa testes de segurança terá sua própria maneira de formatar testes detalhados. Muitas vezes é possível usar o mesmo formato para o projeto de teste de segurança como outros tipos de testes, com a única diferença sendo o alvo do teste e o ambiente de teste.

Mesmo que uma organização siga padrões como IEEE 829-2008 e ISO 29119 [ISO / IEC / IEEE 29119-3], o uso desse padrão deve ser adaptado para atender às necessidades de uma organização. Estes padrões, no entanto, formam um entendimento padrão do que deve ser contido em vários documentos de planejamento de teste. Em muitos casos, os casos de teste e procedimentos de teste (*scripts*) podem ser definidos e implementados em uma ferramenta de gerenciamento de teste que muitas vezes fornece estrutura de formatação.

Casos de teste são a forma mais autônoma de descrição do teste. Eles não exigem execução sequencial. Se a execução sequencial é necessária para atingir um objetivo de teste específico, os casos de teste são combinados em uma sequência expressa em um procedimento de teste ou *script*. Os casos de teste são tipicamente usados para testar condições únicas. Por exemplo, nos testes de segurança, testar a função de *login* pode consistir em casos de teste projetados para validar que os requisitos de formatação de senha são aplicados corretamente.

Durante a implementação do teste, os casos de teste são desenvolvidos, priorizados e organizados na especificação do procedimento de teste. O procedimento de teste especifica a sequência de execução do caso de teste. Se os testes são executados usando uma ferramenta de execução de teste, a sequência de ações é especificada em um script de teste (que é um procedimento de teste automatizado). Os procedimentos de teste são usados quando a sequência é importante. Por exemplo, um procedimento de teste seria útil para testar o processo de "recuperação de senha perdida".

Quando são necessários testes baseados na experiência, tais como testes exploratórios, as condições de teste e os resultados esperados não são definidos antes do teste, mas as condições testadas e os resultados reais devem ser registrados pelo testador em segurança para relato.

3.3.2 Modelagem de teste de segurança baseado em políticas e procedimentos

Ao modelar testes para validar políticas e procedimentos de segurança, esses itens tornam-se a base do teste. Desta perspectiva, os testes de segurança são quase um meio de auditoria de segurança.

Políticas de segurança e procedimentos não devem ser a única base de testes, porque outras perspectivas de testes de segurança são necessárias. Os objetivos da criação de testes para validar políticas e procedimentos de segurança são:

- Compreender o propósito e o escopo
- Avaliar a testabilidade
- Criar testes relacionados diretamente

Por exemplo, pode haver um procedimento que declara: "*Todos os sistemas de TI da XYZ limitam o número de tentativas de login malsucedidas a três. Um período de bloqueio especificado ocorrerá após três tentativas de login sem êxito. As pessoas que não têm as informações de conta de usuário local apropriadas não poderão acessar nosso sistema de TI e devem entrar em contato com os serviços de suporte de TI para verificar a identidade e obter uma senha temporária* ".

Este é um procedimento muito testável, que exigiria as seguintes etapas:

1. Tentativa *login* em um aplicativo três vezes sem êxito. Uma mensagem de bloqueio deve aparecer na terceira tentativa sem êxito. Quaisquer outras tentativas de *login* na conta recebem a mensagem de bloqueio.
2. Entre em contato com os serviços de suporte de TI e verifique a identidade. Uma senha temporária será emitida para um endereço de e-mail conhecido.
3. Faça o *login* com a senha temporária. O acesso deve ser concedido.
4. Crie uma senha que esteja em conformidade com a diretiva de senha. A nova senha deve ser aceita.
5. Sair.
6. Faça *login* com a senha recém-criada. O acesso deve ser concedido.

Observe que a etapa 4 fornece uma oportunidade para testar a diretiva de senha também.

Nem todas as políticas de segurança são testáveis. Por exemplo, "*O conteúdo dos registros de auditoria da XYZ, Inc. contém todos os eventos auditados com carimbo de data e hora e são rastreáveis para indivíduos específicos. Registros específicos do fabricante que fornecem informações suficientes para cumprir esses requisitos devem ser considerados adequados para fins de auditoria*".

Embora não seja impossível testar, um teste precisaria ser definido e executado para cobrir todos os eventos auditados. As ações teriam que ser executadas para acionar um conjunto de exemplos de eventos a serem registrados nos logs de auditoria e a precisão das informações registradas precisaria ser verificada como correta, como o ID do usuário e o carimbo de data e hora.

3.4 Execução do teste de segurança

3.4.1 Elementos-chave e características de um ambiente de teste de segurança

Embora muitas formas de teste possam utilizar um ambiente de teste localizado no mesmo servidor e em rede com outros sistemas, os testes de segurança têm riscos exclusivos que exigem uma abordagem segregada para a construção do ambiente de teste. Isso é especialmente verdadeiro ao testar aplicativos não confiáveis (como de um provedor de terceiros ou de código aberto).

Alguns testes de segurança, como testes de controles funcionais e gerenciamento de sessões, podem ser executados em um ambiente de teste integrado típico sem alto risco. No entanto, ao testar o código desconhecido e não confiável, a possibilidade de um *malware* corromper um servidor ou rede torna aconselhável testar em um ambiente isolado ou virtual.

Os atributos principais para um ambiente de teste de segurança são os seguintes:

1. **Isolado**: de outros sistemas (dependendo do nível de risco de *malware*).
2. **Completo**: o ambiente total terá de refletir o ambiente alvo (produção) em termos de:
   - Sistemas e aplicações em teste.
   - Sistemas operacionais (versão e configuração exatas).
   - Rede.
   - *Middleware*.
   - *Desktops* (marca de hardware, processador, memória).
   - Dispositivos móveis (fabricante, processador, memória, gerenciamento de energia).
   - Bases de dados.
   - Direitos de acesso.
   - Navegadores e *plugins*.
   - Aplicações coexistentes.
   - Dados (dados de teste de engenharia ou dados de produção que foram mascarados).
3. **Restaurável**: repetir os testes conforme necessário e caso ocorra de o ambiente ser corrompido, o mesmo possa ser restaurado.

3.4.2 Importância do planejamento e das aprovações em testes de segurança

Há várias razões pelas quais um testador em segurança deve ter aprovação antes de executar testes de segurança:

- Em quase todos os países, é contra a lei (tentar) obter acesso aos sistemas de dados e suas informações. Em alguns, é contra a lei ter acesso a ferramentas de teste de segurança. Isso significa que na maioria das atividades de teste de segurança você quebrará uma ou mais leis. A única maneira possível de realizar o teste é obter uma renúncia do proprietário do sistema ou dos dados, e aprovação de sua administração.
- Os testes de segurança podem desencadear alertas de detecção de intrusão e o testador pode parecer ser um invasor malicioso. O teste de penetração é um caso específico quando essa autorização é necessária.
- Os testes de segurança podem levar a grandes falhas do sistema e *blackouts*. O risco deve ser conhecido e possíveis precauções devem ser implementadas.

Sem autorização prévia e específica para testes de segurança, um testador pode estar violando políticas e procedimentos de segurança. Isso pode tornar o testador sujeito a rescisão ou acusação.

Um formulário de autorização para testes de segurança deve conter as seguintes informações:

- Nome da entidade autorizadora.
- Nomes do pessoal de teste e/ou entidade.
- Declaração de trabalho.
- Datas de autorização (de/para).
- Outros detalhes relevantes, como endereços IP de origem, contas de usuário e assim por diante.
- Atestados:
  - O cliente possui o sistema a ser testado.
  - O cliente tem autoridade para autorizar testes de segurança.
  - O cliente realizou um backup de todos os sistemas e dados.
  - O cliente testou a restauração do sistema a partir de backups.
  - O cliente entende os riscos associados aos testes de segurança.
- Uma cláusula "inofensiva" para a entidade de teste.
- Assinaturas do representante do cliente autorizado a celebrando os acordos.

Um formulário de amostra pode ser encontrado em [OWASP3].

3.5 Avaliação do teste de segurança

Como grande parte dos testes, a avaliação do teste de segurança é realizada durante a execução à medida que são realizados testes individuais. A avaliação do teste de segurança é a avaliação do resultado de um teste de segurança. Quando os defeitos (vulnerabilidades) são identificados, um relatório de incidente deve ser arquivado, afirmando no mínimo:

- O nome do testador que observou a vulnerabilidade.
- O ambiente de teste onde a vulnerabilidade foi observada.
- As etapas do teste executadas (para facilitar a reprodução dos resultados do teste).
- A natureza da vulnerabilidade da segurança.
- A extensão da vulnerabilidade de segurança.
- O potencial impacto da vulnerabilidade de segurança.
- O plano de ação sugerido para remediação.

Os relatórios de incidentes de testes de segurança podem ser arquivados usando o mesmo sistema de gerenciamento de incidentes que outras formas de teste. Os relatórios de teste de segurança devem ser atribuídos uma categoria especial e podem precisar de proteção para proibir a visualização por pessoal não autorizado. Tais situações existem quando:

- Os testes de segurança estão sendo realizados por uma organização independente e os incidentes são relatados em uma ferramenta que tem poucas restrições na exibição de relatórios de incidentes.
- As vulnerabilidades de segurança podem ser identificadas, mas não imediatamente resolvidas.
- O pessoal interno pode ser considerado uma ameaça potencial para tirar proveito das vulnerabilidades de segurança.

O auditor de TI deve ser capaz restringir ou não o acesso aos resultados dos testes de segurança.

Na conclusão de um grande esforço de teste de segurança, como na conclusão do teste do sistema, um relatório final pode ser emitido. Este relatório também pode ser considerado confidencial, dependendo do status da resolução de vulnerabilidade.

3.6 Manutenção do teste de segurança

Em muitos casos, modificar o processo de teste de segurança pode consistir somente em adicionar novos tipos de testes em resposta a novos tipos de ameaças. Uma coisa é certa, as metas e ameaças de testes de segurança mudam diariamente, portanto, o processo de teste de segurança precisa ser projetado para mudar facilmente.

Novas ferramentas também aparecem no mercado para ajudar a realizar testes de segurança. Os testadores em segurança devem acompanhar esses avanços e avaliar quais ferramentas podem adicionar poder e flexibilidade aos testes de segurança.

4 Teste de segurança durante o ciclo de vida do software (225 min)

Palavras-chave

Caso de uso de abuso, teste *fuzz*.

Objetivos de aprendizagem

4.1 O papel dos testes de segurança no ciclo de vida do software

AS-4.1.1 (K2): Explicar por que a segurança é melhor alcançada dentro de um processo de ciclo de vida.

AS-4.1.2 (K3): Implementar as atividades relacionadas à segurança, apropriadas para um determinado ciclo de vida do software (p.e., iterativo, sequencial)

4.2 O papel dos testes de segurança em requisitos

AS-4.2.1 (K4): Analisar um determinado conjunto de requisitos na perspectiva da segurança para identificar deficiências.

4.3 O papel dos testes de segurança na modelagem

AS-4.3.1 (K4): Analisar um determinado documento de projeto do ponto de vista da segurança para identificar deficiências.

4.4 O papel dos testes de segurança nas atividades de implementação

AS-4.4.1 (K2): Compreender o papel dos testes de segurança durante o teste de componentes.

AS-4.4.2 (K3): Implementar testes de segurança no nível de componente (resumo) dados uma especificação de codificação definida.

AS-4.4.3 (K4): Analisar os resultados de um determinado teste no nível de componente para determinar a adequação do código a partir da perspectiva de segurança.

AS-4.4.4 (K2): Compreender o papel dos testes de segurança durante os testes de integração de componentes.

AS-4.4.5 (K3): Implementar testes de segurança de integração de componentes (resumo) dados uma especificação de sistema definida.

4.5 O papel dos testes de segurança em atividades de teste de sistema e aceitação

AS-4.5.1 (K3): Implementar um cenário de teste de ponta a ponta para testes de segurança que verificam um ou mais requisitos de segurança, testando um processo funcional descrito.

AS-4.5.2 (K3): Demonstrar a capacidade de definir um conjunto de critérios de aceitação para os aspectos de segurança de um dado teste de aceitação.

4.6 O papel dos testes de segurança na manutenção

AS-4.6.1 (K3): Implementar uma abordagem de teste, reteste ou regressão de segurança de ponta a ponta com base em um determinado cenário.

4.1 O papel dos testes de segurança no ciclo de vida de um software

A segurança não é testada ou corrigida em um aplicativo já construído. Em vez disso, é alcançada através de modelagem orientada à segurança e verificação durante todo o processo de construção. Como os testes de software em geral, os testes de segurança também são um processo que devem ocorrer dentro do ciclo de vida do desenvolvimento.

4.1.1 Visão do ciclo de vida dos testes de segurança

Um processo de ciclo de vida do software fornece uma estrutura para executar certas atividades que às vezes se alinham com outras atividades. Por exemplo, as necessidades do usuário devem ser obtidas antes da modelagem do aplicativo ocorrer. A seleção do ciclo de vida do software depende da natureza da organização, do projeto e de fatores similares [IEEE 12207]. Para efeitos deste programa e testes de segurança, os conceitos e técnicas podem ser aplicados a qualquer processo do ciclo de vida - sequencial ou iterativo.

No Capítulo 3 deste programa, foi descrito um processo genérico de teste de segurança que alinha com um ciclo de vida de software. As razões para integrar os testes de segurança no ciclo de vida do software são discutidas nas seções a seguir.

Para fornecer um tempo prescrito no ciclo de vida quando as atividades relacionadas à segurança devem ocorrer, por exemplo, ao capturar e definir as necessidades do usuário, o analista de negócios ou de sistemas deve fazer perguntas como:

- Que níveis de acesso à segurança são necessários?
- Existem ativos digitais ou físicos que exijam defesas especiais de segurança?
- Quão "aberta" a aplicação elencada deve ser?
- Quais são os riscos de segurança?

Outro exemplo seria durante a codificação. Neste momento, o desenvolvedor tem a melhor oportunidade para aplicar práticas seguras de codificação para evitar ataques como, *SQL injection* e ataques de estouro de buffer de memória. Encontrar esses tipos de vulnerabilidades durante etapas posteriores do projeto é difícil e dispendioso, pois muitos outros componentes de software também precisam ser endereçados e corrigidos de maneira semelhante.

4.1.1.1 Fornecer pontos de verificação para revisão

Por exemplo, requisitos de segurança ou estórias de usuários devem ser revistos para garantir que os aspectos relacionados à segurança das necessidades do usuário tenham sido adequadamente investigados e documentados. As alterações de código também devem ser revistas para detectar a presença de códigos maliciosos por parte de funcionários internos ou contratados.

4.1.1.2 Fornecer pontos de verificação para testes

Por exemplo, no desenvolvimento os testes de componentes devem ser documentados e executados para verificar se as práticas de codificação seguras foram seguidas e implementadas com sucesso.

4.1.1.3 Fornecer critérios de entrada e saída ao longo do projeto

Um exemplo desta prática seria que nenhum componente pode ser aceito em um ambiente de teste integrado até que possa ser mostrado que todas as atividades relacionadas à segurança (desenvolvimento e teste) foram concluídas com êxito. Isso é especialmente importante em fases posteriores do projeto, onde uma vulnerabilidade de segurança pode causar um risco de segurança que afeta todo o sistema ou aplicativo.

4.1.2 Atividades relacionadas à segurança no ciclo de vida do software

As seguintes atividades relacionadas à segurança são executadas ao lado de outras atividades do projeto, ao contrário de serem realizadas em seu próprio ciclo de vida separado.

Requisitos: Requisitos são reunidos e definidos em uma variedade de maneiras, dependendo do ciclo de vida do software em uso. Deve-se reconhecer que os requisitos podem ir além das necessidades dos usuários e das partes interessadas. Por exemplo, pode haver requisitos regulamentares, requisitos técnicos e requisitos de negócios, entre outros.

Os objetivos dos requisitos incluem:

- Compreender e identificar as necessidades de segurança de todas as perspectivas dentro da organização e fora da organização. Por exemplo, o cliente de uma empresa não está na organização, mas eles têm a necessidade de suas informações privadas para permanecerem seguras.
- Documentar as necessidades de segurança de forma detalhada e inequívoca. Isso permite a implementação e teste que é rastreável para os requisitos, permitindo que os requisitos sejam verificados e validados.

As atividades de requisitos incluem:

- Definir todas as pessoas afetadas e conhecedoras que possam ter contribuído para os requisitos.
- Utilizar uma variedade de métodos, entrevistas, *workshops*, etc., reunindo as necessidades de segurança expressas por cada grupo. Isto também pode ser realizado durante a elicitação de outros requisitos.
- Documentar os requisitos de uma forma que possam ser revisados e rastreados.
- Revisar os requisitos no que se refere à correção, integridade, compreensão e não ambiguidade.

Modelagem: O sistema ou aplicativo é projetado com base nas necessidades indicadas nos requisitos. Os requisitos expressam as necessidades de segurança enquanto a modelagem traduz as necessidades em uma abordagem de solução viável.

Os objetivos do projeto incluem:

- Criar um projeto de sistema ou aplicativo que atenda aos requisitos de segurança estabelecidos.

As atividades de projeto incluem:

- Analisar os requisitos documentados.
- Chegar à abordagem mais viável para desenvolver a aplicação de forma segura.
- Documentar o projeto usando as técnicas apropriadas de acordo com o ciclo de vida do software. Por exemplo, numa abordagem iterativa, as sessões de projeto podem ser realizadas em um quadro branco, enquanto em outros processos a modelagem pode ser expressa em modelos.

Implementação: Isto é comumente conhecido como a atividade de codificação.

Os objetivos de implementação incluem:

- Traduzir os requisitos e a modelagem em código seguro que atenda às necessidades funcionais conforme indicado nos requisitos.
- Implementar quaisquer outros procedimentos ou tecnologias necessárias (*firewalls*, *tokens*, etc.) para atender às necessidades de segurança.

As atividades de implementação incluem:

- Criar código que atenda aos requisitos de segurança.

- Realizar testes de componentes para verificar a correção, a eficiência e a segurança da implementação.
- Realizar revisões de componentes para inspecionar visualmente a correção, a eficiência e a segurança da implementação

Teste do sistema:

Observe que alguns modelos de ciclo de vida de software, como abordagens de entrega iterativas, adicionam novos componentes ou refinam componentes existentes em um período mais curto de tempo, sendo capazes de ter o teste do sistema com muito mais frequência do que outras abordagens mais sequenciais.

Os objetivos de teste do sistema incluem:

- Realizar um teste de ponta a ponta para observar o funcionamento geral e o desempenho do sistema completo (hardware, software, dados, pessoas e procedimentos) depois dos vários componentes do sistema terem sido implementados e integrados em um sistema completo.
- Testar se os requisitos de segurança foram implementados corretamente do ponto de vista do sistema.

As atividades de teste do sistema incluem:

- Realizar testes de segurança em um ambiente o mais aproximado possível do ambiente de produção, necessitando de uma transição do ambiente de desenvolvimento no qual as atividades anteriores de implementação e integração ocorreram.

Teste de aceitação:

Este é o nível final de teste durante o qual os usuários ou representantes dos usuários do sistema acreditam que o sistema fornecerá os recursos necessários no ambiente de produção.

Os objetivos de teste de aceitação incluem:

- Ter os usuários ou agentes atuando em nome dos usuários, realizando testes de segurança contra os critérios de aceitação relacionados à segurança estabelecidos para o sistema. Muitas vezes, estes critérios se concentram em controles e processos de segurança funcionais.

As atividades de teste de aceitação incluem:

- Instalar o sistema em seu ambiente operacional.
- Realizar testes de segurança com base em critérios de aceitação.
- Determinar a aceitação com base nos resultados dos testes.

Deve notar-se que tanto o sistema como os testes de aceitação são essencialmente testes caixa-preta ou de resposta a estímulos sem considerar a estrutura interna ou o comportamento dos componentes dentro do sistema global. O componente anterior e os testes de integração fornecem avaliações complementares considerando e explorando a arquitetura interna dos componentes e suas interações dentro do sistema.

Manutenção:

Depois de um sistema ter sido colocado em serviço, pode ser necessário um esforço de desenvolvimento adicional para corrigir defeitos na versão lançada (manutenção corretiva), para ajustar a outras mudanças no ambiente operacional (manutenção adaptativa) ou para ampliar ou aprimorar recursos (manutenção aperfeiçoada).

A perspectiva de testes de segurança para a manutenção do sistema centra-se em testar as alterações feitas para corrigir os defeitos (testes de confirmação) e a funcionalidade principal (teste de regressão) para:

- Assegurar que nenhuma nova vulnerabilidade tenha sido introduzida no sistema pelas atividades de manutenção.
- Verificar se as defesas de segurança existentes ainda são eficazes após uma alteração.

Nesse contexto, a manutenção pode incluir atualizações (por exemplo, sistema operacional, bancos de dados), alterações de codificação, conversões de dados e migrações de plataformas.

Em essência, qualquer atividade de manutenção deve ser tratada com o mesmo cuidado e atenção que o desenvolvimento original. Caso contrário, o risco de introduzir novas vulnerabilidades pode comprometer seriamente a segurança do sistema operacional.

4.2 O papel dos testes de segurança em requisitos

As seguintes considerações precisam ser entendidas sobre os requisitos em geral:

- Muitas organizações são desafiadas apenas a escreverem requisitos básicos do usuário que sejam claros, inequívocos, completos, corretos e testáveis.
- Os requisitos são altamente sujeitos a mudanças ao longo de um projeto e, portanto, a manutenção de requisitos pode ser um desafio.
- São necessárias habilidades especiais para entender as necessidades do usuário e outras como a conformidade e necessidades técnicas, antes de poder redigi-las ou inseri-las em ferramentas de gerenciamento de requisitos.
- Os requisitos podem conter lacunas e erros, portanto, tanto a verificação como a validação são necessárias.
- Requisitos devem conter características de qualidade necessárias, como segurança, desempenho, usabilidade e assim por diante. No entanto, esses atributos são muitas vezes ignorados em favor da funcionalidade apenas.

O desafio é conseguir que a perspectiva de segurança seja compreendida e expressa no conjunto completo de requisitos para um projeto. Ao avaliar os requisitos, uma técnica eficaz é usar uma lista de verificação como um guia. Pode haver muitos itens contidos na lista de verificação para cobrir uma variedade de tópicos. Para os atributos relacionados à segurança, o seguinte é um bom ponto de partida para a avaliação:

Necessidades de Privacidade:

- Todos os grupos de usuários e suas necessidades de privacidade de dados foram identificadas e documentadas?
- Foram identificados todos os tipos de dados impactados por esse requisito e definidas as necessidades de privacidade relacionadas?
- Os direitos de acesso dos usuários foram identificados e definidos?

Necessidades de Conformidade (para políticas de segurança):

- Todas as políticas de segurança relevantes foram identificadas e documentadas?
- Foram identificadas e documentadas exceções às políticas de segurança?

Vulnerabilidades Comuns: Estas mudanças serão alteradas ao longo do tempo, à medida que os ataques de segurança mudam, mas devem ser definidas como riscos no momento de se definir os requisitos. Estes também se tornam a base para testes de segurança.

- Todas as vulnerabilidades comuns e conhecidas para o recurso que está sendo documentado foram identificadas como riscos conhecidos?

Testabilidade:

- O requisito está escrito de forma que os testes de segurança e outros testes possam ser escritos com base no documento?
- Quaisquer termos ambíguos como "processamento deve ser seguro" e "acesso apenas é concedido ao pessoal autorizado" foram identificados e esclarecidos para serem específicos e testáveis?

Usabilidade: Há *trade-offs* entre segurança e usabilidade. Por exemplo, um *login* de usuário em um site pode ser tão confuso e difícil que os clientes desistem e vão para outro lugar.

- Os requisitos refletem um nível apropriado de processo de segurança em relação à função especificada?
- Os procedimentos de segurança são claros e compreensíveis?
- Há soluções específicas para usuários legítimos que podem ter problemas no acesso a informações?

Desempenho: Há um *trade-off* entre segurança e desempenho. Por exemplo, é possível que altos níveis de criptografia diminuam o desempenho.

- Os requisitos refletem um nível apropriado de eficiência de segurança em relação à função especificada?

4.3 O papel dos testes de segurança na modelagem

As práticas de modelagem que degradam a segurança devem ser identificadas e evitadas. As atividades relacionadas ao teste contribuem com o reconhecimento de projetos de sistemas de software que provavelmente serão vulneráveis a comprometer e direcionar a modelagem de sistemas de software com propriedades de segurança fortes e identificáveis.

O IEEE *Computer Society Center for Secure Modelagem* [IEEE1] recomenda estas abordagens de modelagem chave:

- Ganhe ou dê, mas nunca assuma, confiança.
- Use um mecanismo de autenticação que não possa ser ignorado ou adulterado.
- Autorizar após a autenticação
- Instruções e dados de controle estritamente separados, nunca processando instruções de controle recebidas de fontes não confiáveis.
- Definir uma abordagem que garanta que todos os dados são explicitamente validados.
- Usar criptografia corretamente.
- Identificar dados sigilosos e como devem ser tratados.
- Considere sempre os usuários.
- Compreender como a integração de componentes externos altera a sua superfície de ataque.
- Seja flexível ao considerar mudanças futuras em objetos e atores.

4.4 O papel dos testes de segurança nas atividades de implementação

Os testes de segurança, como em outros tipos de testes, começam no nível mais baixo de implementação, exercendo componentes de software separados que serão montados no sistema. Após a avaliação estática desses componentes, os testes fornecem um nível adicional de avaliação, examinando o comportamento dinâmico em resposta a entradas válidas e inválidas.

4.4.1 Teste de segurança durante o teste de componentes

4.4.1.1 Considerações sobre teste caixa-branca

Os testes estáticos, envolvendo toda a gama de inspeção, acompanhamento, auditoria e atividades de revisão técnica, já foi observado.

Os chamados testes caixa-branca (estruturais) referem-se a testes concebidos com base na visibilidade do desenho ou implementação do software. Em contraste, o teste caixa-preta (funcional e não funcional) não se baseia no acesso a nenhuma dessas informações estruturais e é simplesmente um teste de resposta a estímulo.

O teste caixa-branca pode direcionar controles específicos implementados dentro do módulo e determinar sua efetividade. A visibilidade na estrutura do componente também permite medir a cobertura do teste, como em termos de percentual de extratos executáveis exercidos, porcentagem de resultados de decisões exercidos ou porcentagem de trajetórias lógicas percorridas.

Os testes de segurança estrutural podem ser realizados por ferramentas automáticas de análise estática e ferramentas de verificação de segurança. O teste *Fuzz* é uma técnica de teste de segurança usada para descobrir vulnerabilidades de segurança, introduzindo quantidades maciças de dados aleatórios, chamados *fuzz*, para o componente ou sistema sob teste. O teste *Fuzz* de caixa-branca (em pequenos blocos de software, funções, classes) pode obter resultados utilizáveis em muito menos tempo do que uma ferramenta de *fuzzing* para testes caixa-preta.

As ferramentas de teste *Fuzz* de caixa-branca são capazes de detectar corrupção de memória, estouro de *buffer*, etc., instrumentando o código que está sendo testado.

As vulnerabilidades de segurança a seguir podem ser identificadas e corrigidas durante os testes estruturais:

- Estouro do buffer de memória.
- Código malicioso inserido por um empregado ou contratado interno.
- Acesso "*Backdoor*" (acesso via uma interface não documentada conhecida apenas pelo desenvolvedor, intencionalmente implementada para ignorar os controles de segurança normais)

4.4.1.2 Considerações sobre testes de segurança funcionais

A adequação dos testes de segurança a qualquer nível deve ser determinada confirmando a satisfação dos requisitos de segurança especificados, além de observar respostas a estresses não explicitamente especificados em requisitos de segurança, avaliações de risco de segurança e documentos similares. A criatividade é necessária para testar os pontos fracos de segurança, porque os testadores estão investigando o que os desenvolvedores de software ignoraram.

4.4.2 Modelagem de teste de segurança no nível do componente

Um exemplo de conjunto de práticas recomendadas de codificação de alto nível pode ser encontrado no artigo "*Top 10 Secure Coding Practices*" [CERT1] que afirma:

"*Os testes para qualquer componente devem incluir a avaliação de possíveis violações destas práticas:*

- *Validar entrada.*
- *Respeitar os avisos do compilador.*
- *Arquitetura e modelagem para políticas de segurança.*
- *Manter a simplicidade.*
- *Negação predefinida.*
- *Aderir ao princípio do privilégio mínimo.*
- *Desinfetar os dados enviados para outros sistemas.*

- *Defesa da prática em profundidade.*
- *Usar técnicas eficazes de garantia de qualidade.*
- *Adotar um padrão de codificação segura.* "

Os testes realizados em relação a essas listas de verificação das melhores práticas devem incluir avaliações de possíveis violações destas práticas com base numa análise de risco bem documentada que incorpore modelos de ameaças realistas. Em outras palavras, concentre-se nos requisitos mais cruciais em termos de probabilidade de ataque e as consequências do compromisso.

4.4.3 Análise de testes de segurança no nível de componente

Uma medida chave da adequação envolve a avaliação da cobertura do teste. Diversas medidas de cobertura derivam da natureza do teste realizado.

Testes baseados em requisitos testam o sistema para fornecer garantia de que ele atende aos requisitos especificados. Sem consideração para a implementação (caixa-preta), a cobertura pode ser medida com qualquer uma das seguintes porcentagens:

- Número total de requisitos testados.
- Casos de uso / abuso especificados testados.
- Funções críticas, cenários ou segmentos de missão testados.

Testes orientados a dados testam o sistema para fornecer a garantia de seu comportamento em um intervalo e combinação de dados de entrada, tentando escolher em poucos valores de teste possíveis, dividindo o espaço de dados em classes de equivalência e selecionando um representante de cada classe (com a expectativa que os elementos desta classe são equivalentes em termos de sua capacidade de detectar falhas). Os critérios de cobertura *Pairwise* e *Nwise* são formas típicas de critérios de cobertura de dados.

Testes baseados em modelos fornecem garantia de cobertura em termos escolha de uma notação de modelagem. Quando o modelo usa uma notação de pré-condição, os critérios podem incluir cobertura de causa-efeito e cobertura de todos os arranjos na pós-condição. Para as notações de modelagem algébrica, a cobertura dos axiomas é um critério típico de cobertura.

Para os modelos baseados na transição, que utilizam gráficos explícitos que contêm nós e arcos, os critérios de cobertura de gráfico incluem porcentagem de nós (estados), porcentagem de transições, porcentagem de pares de transição e porcentagem de ciclos.

Os testes estruturais fornecem garantia baseada na visibilidade e na análise da implementação real. Por simples enumeração, a cobertura de teste é comumente relatada como a porcentagem dos pacotes, classes, métodos, decisões ou linhas de código executável no aplicativo que são executados pelos testes. Este último é referido como cobertura de declaração.

A complexidade ciclomática é uma medida de quantos caminhos independentes diferentes existem através de um elemento e podem ser visualizados em termos de um gráfico de fluxo de controle com nós (pontos de decisão) e arcos (caminhos). O mais forte dos critérios baseados no fluxo de controle é a cobertura do caminho, que mede contra todos os caminhos de entrada a saída no gráfico de fluxo de controle. Uma vez que o teste exaustivo de caminho geralmente não é viável por causa de loops, outros critérios menos rigorosos podem ser expressos em termos de caminhos lógicos selecionados considerados críticos (cobertura do caminho crítico) ou da porcentagem de resultados de decisão exercidos (cobertura de desvio).

4.4.4 Teste de segurança durante o teste de integração de componentes

Como componentes de nível inferior são integrados em subsistemas e, eventualmente o sistema final, as possibilidades de violações de segurança não são simplesmente o somatório das vulnerabilidades em cada um

dos componentes. Em vez disso, novos vetores de ataque tornam-se possíveis devido a interações entre componentes e com elementos maiores do sistema e da organização.

Por outro lado, algumas interações entre componentes podem mitigar ou bloquear possíveis sequências que levam a violações de segurança. Novamente, os testadores em segurança precisam de criatividade na busca de algo que os desenvolvedores ignoraram.

Os testes de integração podem demonstrar a complexidade de um projeto de sistema e a estabilidade de seu comportamento. A abordagem de teste de integração (por exemplo, de cima para baixo ou de baixo para cima) pode afetar o momento de revelar preocupações de segurança ou a necessidade de testes adicionais específicos.

4.4.5 Modelagem de testes de segurança no nível da integração de componentes

Tal como acontece com os testes de componentes, os testes de integração devem ser concebidos com base numa análise de risco bem documentada que incorpore uma modelização de ameaças realistas. Como componentes separados são integrados, note que a plataforma (na forma de simuladores e controladores) pode ser necessária para testar menos caminhos através de um sistema durante a integração. À medida que mais componentes implementados são adicionados ao sistema, esta plataforma é removida de forma incremental, permitindo uma avaliação mais completa da funcionalidade, assim como novos caminhos para vulnerabilidades possam ser explorados.

4.5 O papel dos testes de segurança nos testes de sistema e aceitação

4.5.1 O papel dos testes de segurança nos testes do sistema

O teste do sistema é o primeiro exercício de ponta a ponta dos componentes totalmente integrados. Embora geralmente feito em um ambiente de desenvolvimento, ele deve revelar propriedades emergentes do sistema que não teriam sido observadas antes que a integração fosse concluída. Os requisitos de segurança são normalmente considerados em conjunto com um dos requisitos mais funcionais.

Por exemplo, "*No processo de construção do sistema X não pode ser permitido que y aconteça*". À medida que os testes funcionais são conduzidos, o testador deve investigar as maneiras pelas quais as restrições de segurança podem ser violadas.

Os requisitos funcionais, incluindo os de segurança, normalmente atendem aos imperativos. Outras especificações, como casos de uso, casos de abuso, modelos de processo e modelos de transição de estados descrevem procedimentos que podem ser usados para definir cenários de teste de ponta a ponta para testes de segurança.

4.5.2 O papel dos testes de segurança nos testes de aceitação

Os testes de aceitação distinguem-se dos testes do sistema pelo fato de serem realizados em um ambiente operacional realista, se não, na configuração real em que o sistema se tornará operacional. Tais testes permitem uma avaliação razoável do desempenho e outros comportamentos baseados em interações através de interfaces externas. Ele finalmente coloca o sistema no ambiente em que agentes de ameaças externas estariam buscando encontrar fraquezas no dia-a-dia.

Testes de aceitação devem validar que os objetivos iniciais do projeto foram entregues. Isto é conseguido através da concepção e realização de testes para validar que os critérios de aceitação sejam cumpridos. As necessidades de segurança devem ser documentadas nos critérios de aceitação.

O melhor momento para definir e documentar os critérios de aceitação é antes do desenvolvimento ou da compra do sistema. Portanto, uma compreensão inicial pode ser feita entre o fornecedor e o adquirente,

mesmo se ambos estiverem na mesma organização. Também é comum que os critérios de aceitação mudem ou surjam durante um projeto, portanto, esses critérios devem ser analisados quanto ao seu impacto nos testes de segurança.

No contexto dos testes de segurança, os critérios de aceitação podem ser de natureza global. Por exemplo, podem haver pontos de critérios de aceitação que especifiquem o que é aceitável em termos de segurança geral do sistema. Isso incluiria critérios que são aplicados a todas as funções do sistema, como autenticação de usuário, direitos de usuário, níveis de criptografia, trilhas de auditoria e assim por diante. Em outros casos, podem ser necessários critérios específicos de aceitação de segurança. Por exemplo, algumas funções, como a emissão de pagamentos em excesso de um determinado montante podem exigir duas pessoas para aprovar o pagamento.

4.6 O papel dos testes de segurança na manutenção

O teste de regressão pretende confirmar que todos os comportamentos previamente aceitáveis do sistema permanecem intactos após modificações terem sido feitas. Nos aspectos negativos dos testes de segurança, tal confirmação envolveria a verificação de que o sistema continua a resistir com sucesso as tentativas de derrotar os controles de segurança estabelecidos. Os aprimoramentos na usabilidade ou na eficiência são especialmente propensos a sacrificar os controles de segurança.

Os testes de regressão de segurança devem se concentrar em confirmar a satisfação de todos os requisitos de segurança, bem como em testar novas vulnerabilidades que possam ter sido introduzidas durante as atividades de manutenção.

Os testes de regressão são muitas vezes aplicados com uma coleção de casos de teste que se baseiam em testes de funções individuais. No entanto, para testes de segurança, isso é muitas vezes insuficiente para detectar defeitos de regressão com impacto na segurança. Os cenários de teste de regressão de ponta a ponta são mais robustos e proporcionam um nível mais elevado de confiança de que transações completas podem ser realizadas de forma segura.

Para este tipo de teste de regressão, um conjunto de cenários de teste de segurança deve ser definido e testado sempre que uma alteração é feita no sistema. Tenha em mente que as alterações do sistema podem ser estendidas para incluir hardware, arquivos de configuração, sistemas operacionais, SGBDs, rede, software, e quaisquer outros componentes do sistema. Defeitos de regressão podem aparecer a partir de alterações a qualquer um destes. Alguns defeitos de regressão podem ter um impacto na segurança.

Exemplo de cenários:

Os usuários podem fazer *login* em um site e concluir uma compra de forma segura sem comprometer suas informações pessoais.

Os usuários só são capazes de executar ações definidas em seus direitos de usuário e privilégios. (Um usuário que trabalha no departamento de folha de pagamento pode ser capaz de adicionar um novo funcionário, mas não ter acesso às suas informações bancárias.)

5 Mecanismos de teste de segurança (240 min)

Palavras-chave

Anti-malware, autenticação, autorização, zona desmilitarizada, encriptação, *firewall*, *hashing*, ameaça interna, sistema de detecção de intruso, *malware*, varredura de *malware*, zona de rede, *pharming*, *phishing*, *salting*, endurecimento do sistema, varredura de vulnerabilidade.

Objetivos de aprendizagem

5.1 Endurecimento de sistema

AS-5.1.1 (K2): Compreender o conceito de endurecimento do sistema e seu papel no aprimoramento da segurança.

AS-5.1.2 (K3): Demonstrar como testar a eficácia dos mecanismos comuns de endurecimento do sistema.

5.2 Autenticação e autorização

AS-5.2.1 (K2): Compreender a relação entre autenticação e autorização e como elas são aplicadas na proteção de sistemas de informação.

AS-5.2.2 (K3): Demonstrar como testar a eficácia dos mecanismos comuns de autenticação e autorização.

5.3 Criptografia

AS-5.3.1 (K2): Compreender o conceito de criptografia e como ele é aplicado na proteção de sistemas de informação.

AS-5.3.2 (K3): Demonstrar como testar a eficácia de mecanismos comuns de criptografia.

5.4 Firewalls e zonas de rede

AS-5.4.1 (K2): Compreender o conceito de firewalls e o uso de zonas de rede e como elas são aplicadas na proteção de sistemas de informação.

AS-5.4.2 (K3): Demonstrar como testar a eficácia de implementações de firewall e zonas de rede existentes.

5.5 Detecção de intrusão

AS-5.5.1 (K2): Compreender o conceito de ferramentas de detecção de intrusão e como elas são aplicadas na proteção de sistemas de informação.

AS-5.5.2 (K3): Demonstrar como testar a eficácia das implementações de ferramentas de detecção de intrusão existentes.

5.6 Varredura de *malware*

AS-5.6.1 (K2): Compreender o conceito de ferramentas de varredura de *malware* e como elas são aplicadas na proteção de sistemas de informação.

AS-5.6.2 (K3): Demonstrar como testar a eficácia das implementações de ferramentas de verificação de *malware* existentes.

5.7 Mascaramento de dados

AS-5.7.1 (K2): Compreender o conceito de ferramentas de mascaramento de dados e como elas são aplicadas na proteção de sistemas de informação.

AS-5.7.2 (K3): Demonstrar como testar a eficácia das abordagens de mascaramento de dados.

5.8 Formação

AS-5.8.1 (K2): Compreender o conceito de treinamento de segurança como atividade de ciclo de vida do software e por que é necessário para proteger sistemas de informação.

As-5.8.2 (K3): Demonstrar como testar a eficácia do treinamento em segurança.

5.1 Endurecimento do sistema

Ao longo dos anos uma variedade de mecanismos de segurança surgiu como práticas-chave na obtenção de ativos digitais e físicos. Cada um desses mecanismos pode ser aplicado de várias maneiras - algumas através de ferramentas e infraestrutura, outras através de esforço manual. Nenhum desses mecanismos por si só é suficiente na maioria dos casos para garantir a informação. Cada mecanismo tem suas próprias vantagens e desvantagens.

Os testadores em segurança precisam entender as nuances de cada linha de defesa para que testes adequados possam ser projetados para verificar e validar a sua eficácia. Os testadores em segurança de nível avançado precisam entender as implicações de cada um dos mecanismos descritos neste capítulo para projetar uma arquitetura de teste que fornecerá uma estrutura contínua para testes de segurança.

5.1.1 Compreendendo o endurecimento do sistema

Os sistemas modernos estão se tornando cada vez mais complexos, assim sua superfície de ataque está crescendo continuamente. Vulnerabilidades vêm de erros de projeto (por vulnerabilidades de projeto), defeitos de código-fonte (por vulnerabilidades de construção) ou falta de rigor na configuração desses sistemas (por vulnerabilidades de configuração).

O endurecimento do sistema é o processo passo-a-passo de reduzir a superfície de ataque aplicando uma política de segurança e diferentes camadas de proteção. O principal objetivo é garantir o sistema e reduzir os riscos de comprometimento da segurança.

Dependendo do contexto, o endurecimento pode ser aplicado em diferentes níveis:

- Endurecimento de um componente de software ou hardware.
- Endurecimento de um produto ou aplicação.
- Endurecimento de um sistema.
- Endurecimento de um sistema de sistemas.

As defesas de segurança organizacional e técnica a serem aplicadas devem incluir:

- Remover software desnecessário (pode conter defeitos).
- Remover bibliotecas desnecessárias e ferramentas de desenvolvedor (pode conter defeitos).
- Remover contas / *logins* desnecessários (vetores de ataque).
- Remover aplicativos desnecessários (podem conter defeitos) e serviços de rede (vetores de ataque).
- Remover periféricos e portas de hardware desnecessários (por exemplo, portas USB, leitores de cartões).
- Atualizações rápidas e instalação de atualizações (por exemplo, ativar atualizações automáticas).
- Atualizar as configurações.
- Seguir as regras de codificação (evite vulnerabilidades de construção).
- Configurar o servidor de registro remoto (por exemplo, *remote-syslog*) para que, em caso de compromisso, o invasor somente consiga excluir os arquivos de *log* da máquina comprometida, mas não no servidor de *log* remoto.

Os seguintes mecanismos de segurança devem ser usados:

- Autenticação forte e gerenciamento eficiente da autorização (apenas conceder os direitos necessários para realizar ações para funções específicas).
- Criptografia (comunicação e armazenamento hospedado localmente).
- Firewalls (aplicativos pessoais, de sistema ou web) e zonas de segurança definidas (por exemplo, execução em uma caixa de areia).

- Sistema de detecção de intrusão.
- *Anti-malware* e *anti-spyware.*
- Mascaramento de dados e aplicações (por exemplo, proteção contra engenharia reversa).

O endurecimento do sistema é vital para proteger os ativos restritos de uma organização, mas as regras de segurança devem ser aplicadas no nível correto e equilibradas com a usabilidade do sistema. No extremo dessa relação de compromisso, as proteções são desativadas porque bloqueiam a produtividade da empresa.

5.1.2 Testando a eficácia dos mecanismos de endurecimento do sistema

O teste da eficácia dos mecanismos de endurecimento do sistema pode ser realizado de várias maneiras. Eles dependerão da natureza do sistema ou da aplicação que está sendo "endurecida", da restrição dos ativos protegidos e das ameaças identificadas. O endurecimento do sistema restringe o acesso do sistema às funções certas, abre apenas os serviços necessários e monitora as atualizações de aplicativos. Portanto, para testar a eficácia do endurecimento do sistema, os testes devem ser concebidos de modo que se saiba se os esforços de endurecimento estão funcionando, se foram aplicados nos lugares certos de maneira correta. Também é importante testar proteções de endurecimento do sistema que são muito restritivas e podem ser excessivas em vista dos riscos de segurança.

Alguns testes de endurecimento do sistema podem ser baseados na revisão ou na auditoria, enquanto outros podem ser baseados na capacidade de determinados grupos de usuários, na execução de determinadas ações ou no acesso a determinados dados.

Os testes podem incluir:

- A auditoria da configuração dos servidores de banco de dados e aplicativos para verificar se as senhas-padrão foram alteradas.
- A auditoria da configuração do sistema para identificar serviços desnecessários e portas de rede.
- A verificação de componentes, bibliotecas e versões de aplicativos para verificar se eles não são obsoletos e vulneráveis.

Uma varredura de vulnerabilidade pode ser executada para facilitar as tarefas de avaliação, especialmente se o sistema for complexo (por exemplo, um ambiente multisite). Ferramentas de análise estática podem ser usadas para detectar violações de regras de codificação que podem introduzir vulnerabilidades de construção. Os analisadores orientados para a segurança podem ser particularmente úteis para detectar vulnerabilidades.

5.2 Autenticação e autorização

5.2.1 Relação entre autenticação e autorização

Os ativos sigilosos de uma organização (por exemplo, números de contas bancárias de uma lista de clientes, desenvolvimento de um novo produto, etc.) precisam serem protegidos e acessíveis somente por pessoas autorizadas.

A autenticação é baseada na verificação de um identificador de usuário e um *token* para responder às perguntas:

- *Login*: quem é o usuário?
- Senha: o usuário é realmente quem ele finge ser?

Implementações diferentes de mecanismos de autenticação podem ser usadas dependendo da necessidade de proteção à contra-ataques para interceptar uma autenticação ou roubar uma senha. Estes incluem a detecção de senhas fracas, empregando senhas únicas, impressões digitais, certificados de software, certificados em *tokens* rígidos e meios de autenticação semelhantes.

Dependendo da arquitetura de um sistema, do contexto do aplicativo e das necessidades de uma organização (facilidade de gerenciar *login*/senha), os mecanismos de autenticação podem incluir autenticação local, de servidor, de rede, SSO (*Single Sign-On*) ou de significado similar.

A autorização é utilizada para os seguintes fins:

- Para verificar se o usuário autenticado tem os direitos para executar uma ação (por exemplo, o usuário pode fazer *login* em um servidor, mas não pode modificar seus dados ou um usuário está autorizado a usar um servidor FTP, mas apenas em seu espaço dedicado).
- Determinar que nível de acesso deve ser permitido aos recursos do sistema.

Há um forte vínculo entre autenticação e autorização com base no princípio de que um usuário não autenticado não tem direitos ou direitos restritos no sistema (não está autorizado a manipular dados confidenciais). Por exemplo, no contexto de um site comercial, um usuário não autorizado pode ver a lista de produtos, mas antes de comprar o artigo escolhido, o usuário deve criar uma conta de usuário. O usuário autenticado pode comprar um item, mas não pode executar funções administrativas.

5.2.2 Testando a eficácia dos mecanismos de autenticação e autorização

O objetivo dos atacantes é roubar senhas ou ignorar sistemas para executar ações não autorizadas. Geralmente, eles exploram diferentes tipos de fraquezas: erros de codificação (falta de filtragem de entrada), versão antiga e vulnerável das bibliotecas, erros de configuração do sistema (mantendo senhas padrão, direitos padrão) e senhas fracas (por exemplo, a senha mais usada é "123456").

Uma organização pode ter um conjunto de regras de senha que devem ser seguidas, mas se o usuário não for diligente em manter a senha segura, as regras de senha não farão a diferença. Além disso, as regras de senha devem refletir as boas práticas atuais na definição de senhas. Tais práticas podem ser encontradas nas Diretrizes de Construção de Senhas do Instituto SANS [SANS2].

Os testes para os mecanismos de autenticação e autorização podem incluir:

- Ataques de força bruta através de dicionário de dados para tentar descobrir senhas de usuário. Os primeiros passos podem ser tentar "123456", "111111", data de nascimento, nome do animal de estimação, etc.
- Falta de exploração de filtragem de entrada, por exemplo, para injetar solicitações SQL para ser autenticada sem nenhum *login* e senha conhecidos.
- Introduzir uma URI (*Uniform Resource Identifier*) não autorizada (p.e., “../../” em uma conta FTP) ou URL (*Uniform Resource Locator*) para tentar obter acesso a dados confidenciais (p.e., “http://nome.do.site/admin”).

Outro exemplo pode ser explorar uma vulnerabilidade no sistema alvo (talvez porque não tenha sido atualizado) para causar um comportamento não intencional e geralmente resultando em ganhar o controle do sistema e permitir a escalação de privilégios.

5.3 Criptografia

5.3.1 Entendendo a criptografia

Para evitar a divulgação de dado sigiloso, mesmo que possa ser acessado quando armazenado ou trafegando entre cliente e servidor, um mecanismo de criptografia pode ser usado. *Hashing* e *Salting* são métodos usados durante a criptografia.

A criptografia é um processo de codificação de dados (texto simples) em dados cíclicos (criptografar texto), usando um algoritmo criptográfico e segredos, de tal forma que somente pessoas autorizadas têm o direito

de acessar usando um mecanismo de descriptografia. O segredo é compartilhado e só é conhecido pelos usuários autorizados. O objetivo é ter uma criptografia que seja suficientemente forte para impedir que um invasor, que conseguiu roubar dados criptografados, tenha recuperado o texto sem formatação. O uso de algoritmos criptográficos ajuda a garantir a confidencialidade, a integridade, a disponibilidade de ativos sigilosos e o bloqueio à manipulação.

Protocolos criptográficos podem ser usados para proteger informações:

- Armazenado em um sistema, por exemplo, senhas criptografadas em um banco de dados, unidade lógica criptografada, unidade de disco rígido inteiramente criptografada.
- Durante uma comunicação, por exemplo, e-mail criptografado, protocolo de comunicação criptografado (SSL, TLS).

Os principais e bem conhecidos protocolos criptográficos utilizados são:

- Criptografia simétrica: uso de chave secreta compartilhada.
- Criptografia assimétrica: uso de chave pública e privada.

5.3.2 Testando a eficácia dos mecanismos comuns de criptografia

Alguns mecanismos criptográficos são conhecidos por serem fracos, especialmente devido ao tamanho curto das chaves secretas, ou chaves estáticas. Outros mecanismos são vulneráveis porque ou não são implementados com as melhores práticas ou incorporam defeitos de codificação (como estouro de buffer).

Os testes para mecanismos de criptografia devem incluir:

- Testes para vulnerabilidades de modelagem:
  - Avaliação de que os modos corretos são usados na criptografia simétrica.
  - Verificação de que o tamanho das chaves criptográficas não é muito pequeno (p.e., a partir de 2015, uma chave RSA com menos de 2048 bits é considerada insegura).
  - Validação da validade dos certificados e capacidade de levantar um alerta se o certificado for auto-assinado (SSL-trip pode ser usado para evitar ataques *man-in-the-middle*).
  - Repetição de ataque (p.e., ataque contra protocolos *Wired Equivalent Privacy* - WEP).
  - Ataques contra protocolos criptográficos para verificar seu nível de força [Bittau].
- Testes para vulnerabilidades de construção:
  - Revisões de código (p.e., para verificar que a função padrão *random* () não é usada para gerar números aleatórios (semente) porque o algoritmo aleatório é relativamente fácil de quebrar).
  - *Fuzzing* para explorar comportamentos inesperados.
  - Ataques de tempo (analisando o tempo necessário para executar algoritmos criptográficos).
  - Análise de potência (usada para dispositivos de hardware criptografados).
- Testes de vulnerabilidades de configuração:
  - Avaliação de configuração de protocolo criptográfico (p.e., configuração do servidor TLS), protocolos autorizados do cliente baseado no guia de configuração TLS para administradores).
  - Ordem de cifra TLS no lado do servidor, para ver se existe algum meio para rebaixar ou renegociar a cifra que está sendo usada.
- Testes de envelhecimento para verificar mecanismos de criptografia que podem ter se tornado fracos e propensos a ser quebrados.

5.4 Firewalls e zonas de rede

5.4.1 Compreendendo os firewalls

De acordo com [Chapman 2000], "*um firewall é um componente ou um conjunto de componentes que restringe o acesso entre uma rede protegida e a Internet, ou entre outros conjuntos de redes*". Um firewall implementa e impõe uma política de segurança baseada na definição de autorização e proibição. Um firewall pode ser baseado em host (software executado em um único host que monitora entradas/saídas de aplicativos) ou baseado em rede (software que monitora o tráfego entre redes).

A tarefa principal de um firewall é controlar o tráfego entre diferentes zonas de rede confiáveis, filtrando os dados que trafegam na rede. Desta forma, o tráfego malicioso proveniente de uma zona não confiável é detectado e bloqueado.

Uma zona de rede é uma sub-rede identificada com um nível de confiança definido:

- Internet/zona pública considerada como não confiável.
- Várias zonas de segurança denominadas zonas desmilitarizadas ou DMZ, com diferentes níveis de confiança.
- Uma ou mais redes privadas/internas consideradas como as mais confiáveis.

As zonas de rede são partes da configuração do firewall. São utilizadas para definir os fluxos autorizados entre as diferentes redes. Todo o tráfego proibido é bloqueado.

Normalmente, um firewall filtra a comunicação com base em:

- Endereços e protocolos de origem e de destino (endereços Ethernet ou IP, portas TCP/UDP, etc.).
- Opções de protocolo (fragmentação, TTL, etc.).
- Tamanho dos dados.

Os firewalls de aplicativos da Web (WAFs) também filtram a comunicação com base em:

- Conexões de usuários presentes.
- Filtragem de dados (p.e., usando descrições de padrões).

5.4.2 Testando a eficácia do firewall

Devido ao número de protocolos, às suas diferentes opções e à complexidade das redes a proteger, é difícil configurar um firewall de forma eficiente. Os testes de eficácia do firewall devem incluir:

- Digitalização de portas para verificar se a política de segurança está bem implementada.
- Utilizar pacotes de rede malformados e *fuzzing* de rede para explorar um comportamento inesperado (p.e., um *Denial of Service*)
- Ataques de fragmentação para ignorar recursos de filtragem com o objetivo de realizar o ataque por trás do firewall

Outro exemplo de testes, visando o WAF, é codificar e comprimir dados ou mascará-los para ocultar as informações maliciosas que transmite o ataque.

5.5 Detecção de intrusão

5.5.1 Compreendendo as ferramentas de detecção de intrusão

Cada ano, o número de ataques aumenta. As técnicas de intrusão evoluem rapidamente e nenhum sistema é 100% seguro.

Um Sistema de Detecção de Intrusão (IDS) é um sistema (dispositivo autônomo ou aplicativo) que monitora atividades em diferentes níveis (da rede ao aplicativo, 7 camadas do modelo OSI) para detectar violações da política de segurança. Se forem detectados desvios do comportamento normal, são levantados alertas que podem ser analisados para outras ações (por exemplo, bloqueio de tráfego, caminho virtual).

Em relação à padronização do IDS, o *Internet Engineering Task Force Working Group Intrusion Detection Exchange Format* descreve um modelo de modelagem para um IDS baseado em dois modelos de segurança:

- **Modelo de segurança negativa** (detecção baseada em assinatura ou detecção de lista negra): a regra é "*tudo o que não é proibido explicitamente é permitido*". A detecção de intrusão é baseada em uma lista de ataques conhecidos ou padrões.
- **Modelo de segurança positiva** (detecção baseada em comportamento ou detecção de lista branca): a regra é "*tudo o que não é explicitamente permitido é rejeitado*". A detecção de intrusão é baseada na especificação do comportamento do sistema para proteger, por exemplo, as características de uma entrada numa forma descrita como uma expressão regular. A intrusão é detectada se o comportamento se desviar do padrão esperado pelo sistema. O tráfego confiável pode ser usado para gerar a especificação.

Um IDS difere de um firewall onde, quando o segundo olha para fora no tráfego para parar intrusões, o primeiro analisa intrusões suspeitas e levanta um alerta se elas são confirmadas.

5.5.2 Testando a eficácia das ferramentas de detecção de intrusão

A detecção baseada em cenários é fácil de ignorar, porque apenas os ataques conhecidos são detectados. Os testes podem incluir as seguintes técnicas de evasão:

- Codificação de caracteres ou modificação de dados (p.e., adicionar espaço em branco, fim de linhas, etc.).
- Fragmentação de IP, Segmentação TCP.
- Criptografia, mascaramento.
- Codificação de URL.

A detecção baseada em comportamento gera um número elevado de resultados falso-positivos e resultados falso-negativos. Um resultado falso-negativo é qualquer alerta que deveria ter sido informado, mas não foi. Ele pode ocorrer quando um novo ataque é desenvolvido a um IDS que não está preparado, ou talvez sobre uma regra que foi escrita de tal forma a detectar alguns ataques, mas perder outros. Além disso, a precisão deste método de detecção deve ser considerada. É possível que um invasor desvie o comportamento do IDS do seu normal resultando em uma nova especificação contendo comportamento intrusivo. Assim, este novo tráfego não é considerado anômalo. Os testes complementares devem usar tráfego malicioso para adicionar novas especificações intrusivas consideradas como tráfego autorizado.

Algumas entradas podem ser usadas para definir um conjunto de testes para IDS, como "*Sistema de Detecção de Intrusão de Proteção de Perfil*" [PP-IDS] e "*Web Application Firewall Evaluation Criteria*" [WAFEC].

5.6 Varredura de malware

5.6.1 Compreendendo as ferramentas de varredura de *malware*

O código malicioso pode afetar os servidores e computadores dos usuários finais, fornecendo aos seus criadores os privilégios esperados e os dados sigilosos alvos. O código malicioso é colocado no destino usando diferentes meios, como um e-mail com anexos maliciosos, URLs falsas, execução de código no lado do cliente, etc.

Uma aplicação *anti-malware* é um software usado para analisar, detectar e remover códigos maliciosos recebidos de diferentes fontes, com diferentes alvos de detecção: *malware*, *phishing* e *pharming*.

O recurso de detecção principal usado pelo *anti-malware* é uma estratégia baseada em assinatura. O princípio é pesquisar em um banco de dados os padrões conhecidos de dados que descrevem um pedaço de código suspeito. No entanto, novos *malwares* ou àqueles para os quais a assinatura não está presente no banco de dados não serão detectados e poderão infectar sua vítima. Um mecanismo heurístico muitas vezes é incorporado em um *anti-malware* para identificar pequenas variações de padrões maliciosos conhecidos para ajudar a combater este problema.

5.6.2 Testando a eficácia das ferramentas de varredura de *malware*

Os desenvolvedores de *malware* e *backdoors* usam diferentes técnicas para proteger seu código contra a detecção. Algumas destas técnicas incluem:

- Explorar as funções da biblioteca do sistema usadas por *malware* (p.e., *FindWindow* que pode ser usado para fechar uma aplicação *anti-malware*).
- Mascaramento de *strings* para desativar a compreensão do comportamento de código mal-intencionado (p.e., usando criptografia). Um exemplo pode ser armazenar script Java em um documento PDF. Outra é usar compressão como *Ultimate Packer* para executáveis (UPX).
- Carregamento dinâmico de funções e bibliotecas (p.e., para limitar a análise do código malicioso).
- Atualização automática de aplicativos (p.e., *Trojan Skype*).

O *malware* também pode usar outros recursos de hardware, como a Unidade de Processamento Gráfico (GPU), para descompactar código malicioso e armazená-lo na memória para ser executado pelo processador. Nesse caso, o *malware* não pode ser analisado antes de sua execução.

Do ponto de vista de testes funcionais, uma ferramenta como "*Eicar*" (arquivo de teste *anti-malware*) pode ser usada para testar a eficácia do *anti-malware* sem desenvolver códigos maliciosos.

Uma consideração importante ao implementar um novo aplicativo *anti-malware* ou atualizar um aplicativo *anti-malware* existente é testar a implementação em uma plataforma representativa antes de implantá-la em toda a organização. Houve casos em que o software *anti-malware* identificou falsamente arquivos legítimos do sistema operacional como *malware* e os colocou em quarentena, encerrando assim toda a capacidade de computação da organização.

5.7 Mascaramento de dados

5.7.1 Compreendendo a mascaramento de dados

O mascaramento de dados é um mecanismo para tornar os dados e código-fonte não compreensíveis para um ser humano.

Esta técnica é usada principalmente para proteger dados sigilosos contra:

- Cópia, para ignorar mecanismos de proteção de licença.
- Engenharia Reversa, para entender o código para explorar vulnerabilidades.

O mascaramento de dados também pode ser usado para permitir que os funcionários de uma empresa (pessoal de suporte, testadores funcionais, etc.) trabalhem com dados, escondendo os dados sigilosos de vista. Alguns podem se referir a mascaramento de dados como "anonimização de dados", na medida em que mantém anônimos os dados pessoais de um indivíduo.

O mascaramento também pode ser usado para proteger o código-fonte contra o simples copiar-colar (p.e., para proteger um novo algoritmo inovador) e sua reutilização depois de ter sido modificado para entendê-lo.

Às vezes, os desenvolvedores precisam otimizar seu código para torná-los mais eficientes. Isto pode resultar em código fonte mascarado (p.e., codificando algumas partes na linguagem de montagem). Alguns ataques de nível de aplicativo da web consistem em *script injection*. Para ter sucesso, os atacantes precisam conhecer a estrutura do site e das páginas HTML. A mascaramento pode ajudar na proteção de páginas HTML confidenciais e críticas (p.e., páginas de conexão e administração).

Várias técnicas de mascaramento podem ser usadas, tais como codificação de *base64*, *XORing*, renomeando aleatoriamente as funções, substituindo métodos, suprimindo espaço de retorno de tabulação, embaralhamento, etc.. A criptografia também é uma técnica de mascaramento, mas com problemas porque os dados criptografados permanecerão visíveis para àqueles com chaves válidas.

Observação: O mascaramento de dados é frequentemente usado por atacantes para ocultar seus códigos maliciosos e ataques.

5.7.2 Testando a eficácia dos métodos de mascaramento de dados

O controle de configuração entre os dados mascarados e das chaves usadas para a mascaramento é necessário para garantir que as versões corretas das chaves sejam usadas. Caso contrário, os dados não podem ser desmascarados para uso.

Como os dados privados podem estar envolvidos em alguns testes, o seu mascaramento pode ser usado para fins de teste para tornar os dados de produção usados em um ambiente de teste como anônimos. Os dados sigilosos, como informações de usuário usadas por um sistema de informações de saúde, não devem ser divulgados aos testadores. Os testes podem incluir:

- Força bruta ou ataques de dicionário para tentar obter dados simples de dados mascarados.

Os testes para verificar o mascaramento do código podem incluir:

- Engenharia reversa do código Java, por exemplo, regenerar o código-fonte usando o *Java Decompiler* ou programas .Net, por exemplo, para recuperar o código fonte .Net com o *.NET Reflector*.
- Ataques de força bruta, porque alguns mecanismos de mascaramento são vulneráveis, por exemplo, usando *unXOR* [Chopitea].

Em teoria, o código não pode se proteger contra o mascaramento, porque a depuração pode sempre ser usada. Embora existam ferramentas com a finalidade de proteger o código contra a descompilação, ainda existem riscos e limitações na proteção de informações proprietárias representadas por código.

5.8 Formação

5.8.1 Importância do treinamento em segurança

Os seres humanos são muitas vezes o elo mais fraco no quadro geral de segurança. Portanto, é necessário treinamento consistente e contínuo para lembrá-los sobre a importância de seguir as políticas de segurança estabelecidas e para enfatizar porque as políticas são necessárias. Esse treinamento deve ocorrer ao longo do processo do ciclo de vida do software e ser atualizado à medida que novas políticas são adicionadas e novas ameaças surgem. O treinamento deve abranger a identificação de ataques de engenharia social e ameaças internas.

5.8.2 Como testar a eficácia do treinamento de segurança

Por exemplo, um programa de treinamento de segurança pode abordar a importância de ter senhas de usuário fortes que são mantidas confidenciais.

Os testes podem incluir:

- Engenharia social para tentar que um usuário revele sua senha durante uma conversa telefônica com uma pessoa de suporte técnico falso.
- Olhando ao redor de mesas para identificar notas (*postits*) com senhas neles (especialmente sob teclados).
- Executar ferramentas de auditoria de senha para identificar senhas fracas. Um risco deste tipo de ferramenta é que as senhas podem ser visualizadas para a pessoa que executa o teste.

Outro exemplo seria que um desenvolvedor não consegue colocar uma edição em um campo de entrada de dados para impedir a entrada de comandos SQL. Devido a esse erro, um testador em segurança é capaz de injetar um comando SQL e ver o conteúdo de um banco de dados de clientes. Isso indicaria que o desenvolvedor precisa de treinamento adicional em práticas seguras de codificação. Também seria bom examinar as práticas de codificação de outros desenvolvedores para ver se essa prática é comum, assim, uma iniciativa geral de melhoria de processos é necessária.

Um terceiro exemplo seria quando um testador tenta obter acesso físico não autorizado a um escritório e visualizar documentos que foram deixados em aberto.

6 Fatores humanos em teste de segurança (105 min)

Palavras-chave

Ataque, *botnet*, computação forense, *hacker*, reconhecimento, *script kiddies*

Objetivos de aprendizagem

6.1 Compreendendo os atacantes

AS-6.1.1 (K2): Explicar como o comportamento humano pode levar a riscos de segurança e como ele afeta a eficácia dos testes de segurança.

AS-6.1.2 (K3): Para um determinado cenário, demonstre a capacidade de identificar formas pelas quais um invasor pode descobrir informações-chave sobre um alvo e aplicar medidas para proteger o ambiente.

AS-6.1.3 (K2): Explicar as motivações e origens comuns para a execução de ataques de sistemas informáticos.

AS-6.1.4 (K4): Analisar um cenário de ataque (ataque executado e descoberto) e identificar possíveis fontes e motivação para o ataque.

6.2 Engenharia social

AS-6.2.1 (K2): Explicar como as defesas de segurança podem ser comprometidas pela engenharia social.

6.3 Consciência de segurança

AS-6.3.1 (K2): Compreender a importância da sensibilização para a segurança em toda a organização.

AS-6.3.2 (K3): Tendo em conta determinados resultados de teste, aplicar medidas adequadas para aumentar a sensibilização para a segurança.

6.1 Compreendendo os atacantes

No contexto da segurança da informação, o ser humano é tanto a maior ameaça quanto o ponto mais fraco da defesa.

Ataques de segurança são realizados por pessoas com uma variedade de habilidades e motivações. Além disso, os seres humanos são os maiores ativadores para a maioria dos ataques de segurança. Compreender a tecnologia de segurança e implementá-la não é suficiente para se defender de contra-ataques. Também é importante compreender a mentalidade, motivações e métodos dos atacantes maliciosos e estar ciente das fraquezas humanas na linha de defesa.

6.1.1 O Impacto do comportamento humano nos riscos de segurança

A fase chave em qualquer ataque é a fase de coleta da informação (reconhecimento) onde o atacante tenta encontrar e recolher informações sobre o alvo. Toda a informação que é publicada, às vezes sem que se saiba, sobre uma organização, sistemas em uso, etc., e armazenada na Internet será encontrada e poderá ou será usada em um ataque. Não é uma questão de "se", mas uma questão de "quando". Além das informações publicadas oficialmente pela organização, os funcionários também estão publicando informações sobre a empresa em suas redes sociais. A quantidade e o conteúdo desta informação estão em constante mudança, muitas vezes apresentando informações-chave para os atacantes.

Os atacantes não usam uma política de segurança ou procedimentos predefinidos quando atacam um sistema. Com base nas informações que eles podem coletar, eles decidem sua estratégia. Eles atualizarão sua base de conhecimento para cada ataque realizando buscas seletivas e 'visitando' endereços IP já conhecidos.

Quando a política de segurança para uma empresa é formulada, geralmente é baseada na situação e os fatos que estão disponíveis. Às vezes, isso não inclui todas as informações acessíveis ao público e, mesmo que isso aconteça, essas informações provavelmente mudarão. Testes de segurança que eram válidos quando foram criados podem não fornecer cobertura adequada quando as alterações de informações forem publicadas.

6.1.2 Entendendo a mentalidade do atacante

Durante a atividade de reconhecimento ou coleta, o atacante tentará encontrar todos os tipos de informações sobre o alvo usando meios passivos e/ou ativos. A maioria dos equipamentos de TI que trafega em redes públicas deixa uma pegada nessas redes. Essas pegadas podem e serão encontradas. O Google (incluindo o *Google Earth* e *Street View*) ou outros motores de busca, *Shodan* [Web-5], *Facebook*, *LinkedIn* e outras redes sociais são as primeiras fontes usadas para encontrar informações sobre o alvo. Endereços IP, páginas da Web, números de telefone, nomes e estruturas de endereços de e-mail, SO e aplicativos podem fornecer informações úteis ao invasor.

Um possível uso do mecanismo de busca do Google é encontrar informações específicas sobre um destino. Centenas de consultas podem ser encontradas no *Google Hacking Database* [Web-4]. *Shodan* [Web-5] é outra ferramenta que é usada para encontrar informações específicas, por exemplo, quais empresas em uma área específica estão executando um servidor Apache com uma versão vulnerável.

A maioria dessas informações pode ser encontrada passivamente sem realmente se conectar ao sistema de destino. Outras ferramentas utilizadas incluem:

- *Whois* [Web-13].
- Base de Dados Ripe (Redes IP Europeias) [Web-12].
- Pesquisas DNS [Web-25].

Com técnicas de reconhecimento ativas o atacante usa ferramentas para detectar *hosts*, portas abertas, sistemas operacionais e aplicativos conectando no sistema. Métodos e ferramentas usados aqui incluem:

- Ping - Fping [Web-15], Hping [Web-19].
- Varredura TCP / UDP - Nmap [Web-20], Zenmap [Web-21].
- Detecção de SO - Nmap [Web-20], Xprobe2 [Web-22].
- Serviços de impressão de digitais (o Nmap tem capacidade para determinar também o tipo e a versão do serviço em execução na porta aberta descoberta, comparando a "impressão digital" do serviço descoberto com o próprio banco de dados de impressões digitais do Nmap).

Como *hacking* em um sistema é proibido por lei na maioria, se não em todos os países, o hacker vai tentar destruir todas as provas de sua invasão depois. Outras razões para destruir evidências são prolongar a permanência, continuar o uso do sistema no futuro e usar o sistema comprometido ou rede de sistemas (*botnets*) para atacar outros sistemas. O atacante pode implementar ferramentas como o NetCat [Web-14] para isso ou usar sites como o IP TRacer [Web-7], bem como tunelar ou alterar arquivos de log.

Outros métodos e ferramentas utilizados para ocultar evidências incluem ferramentas de ocultação [Web-16], *rootkits* e fluxo de arquivos. Todas, ou a maioria das ferramentas aqui mencionadas são acessíveis através da Internet. Baixando a versão mais recente do Kali Linux [Web-17] e uma pesquisa no site OWASP [OWASP1] dará acesso a muitas dessas ferramentas.

6.1.3 Motivações comuns e fontes de ataques a sistemas de informação

Muitos ataques e violações nos sistemas de informação vêm de dentro da organização. Os usuários de sistemas mal-intencionados (hackers internos ou ameaças internas) tentarão comprometer os sistemas enquanto forem usuários autorizados da rede. Na maioria das vezes a vingança é a motivação, mas tendências recentes estão mostrando um aumento para espionagem econômica ou roubo.

Os hackers externos são responsáveis pela minoria dos ataques. A curiosidade para se ter a informação foi um dos primeiros motivadores para invadir sistemas de informação, e ainda é. Ter alguma informação de grandes empresas ou organizações e saber que outros não o fazem é outro motivador (prestígio). Outros motivadores incluem notoriedade ou fama, desafio, tédio e vingança, onde este último é considerado a forma mais perigosa (maior motivação).

Atacantes são frequentemente classificados por sua motivação e habilidades. Na extremidade inferior do espectro de atacantes estão "*script kiddies*" que simplesmente executam os ataques que outros criam, enquanto na extremidade superior do espectro são profissionais (governo, hacktivismo), organizações e indivíduos. Hacktivismo é o ataque de sistemas baseados em motivos principalmente políticos, mas também econômicos ou demográficos.

A motivação pode escalar para diversão em derrubar um sistema ou organização completamente por qualquer motivo (p.e., político, ideológico, econômico, guerra, comercial, terrorismo).

As habilidades de *hacking* variam de indivíduos com algum conhecimento de rede e sistema trabalhando com um computador doméstico simples, para profissionais altamente treinados e educados com acesso a laboratórios, redes de proxy e todos os outros equipamentos técnicos necessários. Ter uma imagem sobre os atacantes em potencial ajudará uma organização a implementar a proteção necessária e fornece uma diretriz para a estratégia de teste de segurança.

6.1.4 Compreendendo cenários de ataque e motivações

Um incidente de segurança é definido como um evento relevante à segurança de um sistema em que sua a política de segurança é desobedecida ou violada. [RFC2828]

Descobrir o que aconteceu e quem foi responsável pelo incidente relacionado à segurança é um alvo para a disciplina de computação forense [Web-8], onde se concentra em encontrar provas digitais do ataque.

O processo de recuperação de evidências baseia-se em três fases:

1. Adquirir e autenticar.
2. Analisar.
3. Relatório.

6.1.4.1 Adquirir e autenticar

O processo de gerenciamento de incidentes da organização deve restaurar o sistema para seu estado original (pré-ataque) após a evidência ser coletada e armazenada. Ele inicia quando o administrador do sistema é alertado pelo IDS ou por outros meios de monitoramento. Outros sintomas típicos de incidentes de segurança são:

- Entradas de registro suspeitas.
- Contas de usuário inexplicáveis.
- Arquivos ou pastas modificadas.
- Serviços incomuns em execução.
- Comportamento incomum do sistema.
- Tentativas de *login* malsucedidas.

Após ser alertado, o processo a ser executado é o seguinte:

1. Faça um instantâneo ou cópia do sistema em investigação para recolher todas as provas necessárias.
2. Depois de autenticar a evidência (em uma cópia genuína e completa), criar uma cópia e armazená-lo em um local seguro.
3. Analise as evidências.
4. Depois que o processo forense estiver concluído, remova a causa do incidente (erradicação).
5. O sistema volta ao seu estado normal (recuperação).

Durante essas etapas, todas as vulnerabilidades são removidas com patches ou instalando novos softwares. Ao relatar os resultados, o processo seguido deve ser descrito juntamente com as ferramentas usadas durante este processo.

6.1.4.2 Analisar

Após tentativas de *hacking*, pode ser possível encontrar a origem dos ataques examinando os arquivos de *log* do sistema e as conexões de rede ativas. É importante fazer cópias de todos os arquivos de *log* e capturar as informações de status do processo. Durante um ataque ativo, pode fazer sentido reunir informações do sistema relacionadas ao atacante antes de bloqueá-las.

Qualquer ataque via Internet pode ser rastreado para o endereço IP de sua origem, se ele usou e-mail ou conexões de Internet. É apenas uma questão de tempo, dinheiro e esforço, e uma avaliação dos custos envolvidos. A maioria dos atacantes usa proxies ou cadeias de proxies, rede Tor [Web-9] ou outras opções anônimas gratuitas para cobrir seu endereço IP real. Quanto mais proxies os atacantes usam, mais tempo é necessário para rastrear o endereço de origem. As leis locais baseadas nas localizações físicas dos proxies também podem obstruir esta investigação.

Descobrir intrusos e traçar um endereço IP de volta à sua origem pode ser feito com ferramentas como *Netstat* (Windows) [Web-10], *Tracert* [Web-11] e o site *Web Tracer IP* [Web-7]. *Netstat* mostra as conexões para uma máquina, portas e serviços em execução. Esta ferramenta pode ser usada para procurar qualquer endereço IP estranho ou desconhecido ou número de porta. Nota: Há também um utilitário *tracert* no Microsoft Windows (no Linux e OS/X, é "*traceroute*"), mas os serviços baseados na web mencionados acima são independentes desses utilitários.

No cabeçalho de um e-mail contendo vírus, o endereço IP do ISP que enviou o e-mail pode ser mostrado. No entanto, para a maioria dos e-mails baseados na web (Gmail, Yahoo mail, Outlook.com), este é o endereço IP do provedor. Para encontrar o endereço IP real, é preciso procurar o valor *X-Originating-IP*. Usando os bancos de dados *Whois* [Web-13] levará aos detalhes que podem ser usados para entrar em contato com a organização ISP para continuar a investigação. Deve-se notar que o e-mail pode ser originado de servidores privados e servidores de e-mail de retransmissão abertos. Nesse caso, pode ser muito difícil identificar a fonte real de uma mensagem de e-mail.

Investigar ataques que usaram uma *botnet* é difícil. Não há necessidade de o atacante ter uma conexão on-line com o *botserver* ou os *botclients,* é muito difícil ou quase impossível o rastreamento. Neste caso, investigar os clientes pode levar ao servidor, mas é preciso ter acesso ao servidor para investigar a verdadeira fonte do ataque. Os proprietários desses servidores podem não estar cientes de que suas máquinas fazem parte de uma *botnet*.

6.1.4.3 Relatório

O relatório de vulnerabilidades de segurança é descrito no Capítulo 7.

6.2 Engenharia social

Podemos implementar todas as defesas técnicas que imagináveis para proteger os ativos digitais do mundo exterior, mas no final tudo se resume ao fato de que os funcionários (usuários e administradores) precisam ter acesso a esses ativos para fazer seu trabalho. Eles podem precisar usar autenticação para obter acesso de seus desktops, notebooks, telefones inteligentes, *tablets* ou outros meios. Qualquer defesa de segurança física para proteger o acesso ao escritório e equipamento de escritório não tem sentido se a segurança para a área de trabalho do gerente de TI em sua casa pode ser facilmente comprometida.

É o ser humano e o seu comportamento que é a maior ameaça à segurança. Se as pessoas são desleixadas com informações sigilosas, isso deixa muitas pegadas para proteger lugares e transmite essas informações em voz alta (oralmente e eletronicamente) em locais públicos.

A engenharia social é a arte de explorar o ser humano usando seu comportamento geral como vetor de ataque. Como seres sociais, as pessoas estão dispostas a confiar e ajudar estranhos. Isso cria uma vulnerabilidade ao ataque. Ao manipular, influenciar e persuadir pessoas úteis, um invasor tentará obter acesso, detalhes de autorização ou outros tipos de informações confidenciais.

Os *exploits* podem ser realizados por meio de interação humana direta ou usando equipamentos de computador ou rede.

A interação humana direta pode ser feita em pessoa, incluindo:

- *Tailgating* ou *piggybacking* (alguém que não tem a autenticação adequada seguindo um funcionário até uma área restrita).
- Escutar (ouvir as conversas privadas de outra pessoa sem o seu conhecimento).
- Surfar no ombro (olhar por cima do ombro de alguém sem o seu conhecimento enquanto executam tarefas no computador ou por escrito).
- Usar o telefone (como obter uma senha de um usuário desavisado, agindo como outra pessoa, como um gerente ou pessoa de suporte técnico).

A engenharia social baseada em computador pode ser feita por:

- Enviar e-mails infectados com *malware*.

- Usando bate-papo ou aplicações de mensagens instantâneas. Toda a pessoa anônima pode ter um bate-papo com o outro em qualquer lugar no mundo, sem saber a identidade verdadeira da outra pessoa. Além disso, os dados através de mensageiros instantâneos podem ser facilmente rastreados.
- Utilizar telas pop-up. Por exemplo, uma janela pode aparecer na tela do computador do usuário com uma mensagem para o usuário que a conexão de rede foi perdida. Nesse ponto, o usuário é solicitado a digitar novamente seu nome de usuário e senha. Um programa previamente instalado pelo intruso pode então transmitir as informações para um site remoto.
- Enviando e-mails de spam. Os e-mails de spam estão repletos de ofertas e links fraudulentos. Clicar nesses links pode instalar *malware* que pode expor uma rede inteira.
- Persuadir as pessoas a visitar sites infectados (manipulados). Essas tentativas de *phishing* podem ser amplamente enviadas, ou podem ser altamente específicas.

Não existe uma única defesa contra a engenharia social. As defesas podem ser implementadas para controlar os danos (por exemplo, fornecer o menor nível de privilégio que ainda permite que alguém execute seu trabalho, separação de deveres, rotatividade de deveres), mas a principal defesa é a educação e a conscientização em todos os níveis da organização.

6.3 Consciência de segurança

6.3.1 A Importância da consciência de segurança

O modelo de ameaça está em constante mudança, como mencionado em outros capítulos neste *syllabus*. As redes estão evoluindo, novas aplicações são introduzidas, novas interfaces tornam-se operacionais e novas vulnerabilidades são introduzidas e descobertas.

Além desses aspectos técnicos, há o fator humano. Riscos que foram identificados, mas não se tornaram problemas são facilmente esquecidos e as salvaguardas são retiradas. Isso oferece maiores oportunidades para ataques de hackers e engenharia social. O treinamento regular de conscientização de segurança é necessário para manter os administradores e todos os funcionários alertas e informados sobre as mudanças no modelo de ameaça. O treinamento de conscientização de segurança pode se concentrar em diferentes grupos de usuários: desenvolvedores, operações, gerenciamento e equipe geral de usuários.

6.3.2 Aumentar a sensibilização para a segurança

É importante manter uma mentalidade de "consciência de segurança". Além de informações gerais sobre defesas de segurança na empresa, o treinamento deve conter estudos de caso reais descobertos durante os testes de segurança ou em incidentes reais. Com base nesses casos, deve ser mais fácil discutir quaisquer defesas ou mudanças a serem implementadas na organização.

Um esboço para esta seção no treinamento de conscientização deve incluir respostas para as seguintes perguntas:

- Como eles (nós) fizeram isso?
- Quais foram as consequências do negócio?
- Quais foram os custos para investigar e processar o incidente?
- Quais foram os custos para reparar o problema?
- Como o incidente pode ter sido evitado?
- Que mudanças serão implementadas?

7 Avaliação e relatórios de testes de segurança (70 min)

Palavras-chave

Critério de aceite, vetor de ataque, *dashboard*, critério de saída.

Objetivos de aprendizagem

7.1 Avaliação do teste de segurança

AS-7.1.1 (K2): Compreender a necessidade de rever as expectativas de segurança e os critérios de aceitação à medida que o escopo e as metas de um projeto evoluem.

7.2 Relatórios de teste de segurança

AS-7.2.1 (K2): Compreender a importância de manter os resultados dos testes de segurança confidenciais e seguros.

AS-7.2.2 (K2): Entenda a necessidade de criar controles adequados e mecanismos de coleta de dados para fornecer os dados de origem dos relatórios de status do teste de segurança de forma oportuna e precisa (p.e., um painel de teste de segurança).

AS-7.2.3 (K4): Analisar um determinado relatório de status de teste de segurança provisório para determinar o nível de exatidão, compreensão e apropriação de partes interessadas.

7.1 Avaliação do teste de segurança

A mensuração dos resultados dos testes de segurança e o estado de avaliação com relação às expectativas de segurança, critérios de saída e/ou critérios de aceitação são necessários para determinar a conclusão do teste.

É difícil conhecer todos os riscos de segurança no início de um projeto. Além disso, as expectativas das partes interessadas e dos usuários às vezes mudam em relação ao nível de segurança necessário. Por exemplo, a conscientização de uma nova ameaça pode levar as partes interessadas a exigirem níveis mais elevados de segurança do que se pensava inicialmente. Esta é uma razão pela qual as avaliações de risco de segurança precisam ser revisadas ao longo de um projeto e os resultados incorporados no planejamento e execução de testes de segurança.

7.2 Relatórios de teste de segurança

7.2.1 Confidencialidade dos resultados dos testes de segurança

É verdade que o testador médio sabe mais sobre o objeto de teste após o teste ser concluído em comparação com a maioria dos desenvolvedores ou modeladores. Ao testar completamente pode-se encontrar as fraquezas mais importantes e pontos fortes do sistema. O mesmo se aplica aos testes de segurança.

Testando a implementação de segurança, é possível encontrar buracos ocultos e vulnerabilidades de segurança. A diferença reside no possível impacto negativo na comunicação destas vulnerabilidades para outras pessoas que não as partes interessadas diretas. Uma boa prática geralmente é que as informações devem ser disponibilizadas apenas para aqueles que precisam saber. Isso se aplica especificamente aos resultados dos testes de segurança, para ser conservador com a partilha deste tipo de informação é considerada uma boa prática.

7.2.2 Criando controles e mecanismos de coleta de dados para relatórios de teste de segurança

O impacto e o efeito de uma vulnerabilidade de segurança são normalmente considerados como tendo uma sensibilidade mais elevada em comparação com defeitos "normais". Isto leva à necessidade de ser mais preciso e acurado sobre o relato da natureza do defeito e dos riscos implícitos. Na maioria dos projetos, os defeitos de segurança são categorizados com uma gravidade maior do que defeitos funcionais comparáveis.

Esta última implica que a administração tem maior foco nos defeitos de segurança, seus riscos e possíveis resoluções. O relatório de defeitos de segurança deve avaliar cuidadosamente o impacto de um problema descoberto, a precisão dos resultados dos testes deve estar disponível de forma bem definida e oportuna. É uma boa prática discutir com a gerência como e quando eles gostariam de ter acesso aos relatórios de defeitos de segurança.

7.2.3 Analisando relatórios de status de teste de segurança provisórios

Os relatórios de testes de segurança podem ser produzidos durante todo o processo de teste de segurança ou somente no final dos testes de segurança (como no final dos testes de segurança do sistema ou no final dos testes de segurança realizados como parte dos testes de aceitação). Os primeiros relatórios de teste de segurança são incentivados porque permitem mais tempo para remediar vulnerabilidades de segurança. Se o processo de teste de segurança estiver seguindo o descrito neste programa, a equipe de teste poderá descobrir vulnerabilidades e observações de documentos durante todas as atividades de teste.

A estrutura de um relatório de teste de segurança deve conter as seguintes seções:

1. Identificador do relatório
2. Resumo
    a. Sumário executivo
    b. Principais conclusões
3. Desvios
    a. Processo de teste seguido
    b. Qualquer desvio do processo de teste planejado
    c. Métodos e ferramentas (configurações, políticas) utilizadas
4. Avaliação abrangente
    a. Avaliação da cobertura do teste com base nos critérios indicados no plano de teste
    b. Explicação de quaisquer itens ou recursos que não foram testados
5. Resumo dos resultados
    a. Resumindo os resultados dos testes de segurança
    b. Lista de todas as vulnerabilidades de segurança resolvidas e suas resoluções
    c. Lista de todas as vulnerabilidades não resolvidas
6. Avaliação
    a. Avaliação dos resultados dos testes observados e seu status com base em critérios de saída
    b. Riscos identificados (classificações) e impacto de vulnerabilidades de segurança não resolvidas
7. Resumo das atividades
8. Aprovações

A eficácia dos relatórios de testes de segurança depende do seguinte:

- O momento do relatório.
- O conteúdo do relatório.
- Os destinatários do relatório.
- O detalhamento do conteúdo para corresponder à necessidade dos destinatários de múltiplos relatórios pode ser importante para atender às necessidades de várias partes interessadas. Por exemplo, o conteúdo de um relatório para gerenciamento executivo não será o mesmo que o conteúdo para um arquiteto de sistema.

8 Ferramentas de teste de segurança (55 min)

Palavras-chave

Nenhum

Objetivos de aprendizagem

8.1 Tipos e objetivos das ferramentas de teste de segurança

AS-8.1.1 (K2): Explicar o papel das ferramentas de análise estática e dinâmica nos testes de segurança.

8.2 Seleção da ferramenta

AS-8.2.1 (K4): Analisar e documentar como os testes de segurança devem ser abordados por uma ou mais ferramentas.

AS-8.2.2 (K2): Compreender os problemas com as ferramentas de código aberto.

AS-8.2.3 (K2): Compreender a necessidade de avaliar os recursos do fornecedor para atualizar com frequência as ferramentas em uma base para se manter, adequado contra novas ameaças de segurança.

8.1 Tipos e objetivos das ferramentas de teste de segurança

As façanhas desenvolvidas pela comunidade de hackers impulsionaram o desenvolvimento de ferramentas de teste de segurança para defesa contra essas ameaças. Mesmo a partir das primeiras atividades de *hacking* (como *cracking* de senhas) ferramentas simples foram inventadas, criadas e melhoradas por aqueles que as usam. Ferramentas que provaram ser eficazes foram compartilhadas na comunidade de hackers e melhoradas e aprimoradas. Inicialmente, essas ferramentas foram desenvolvidas para tarefas em ambientes dedicados. A Usabilidade não foi um problema, pois quase todos os usuários tinham um fundo técnico. Eventualmente, algumas das ferramentas de hackers se tornaram a base para ferramentas de teste de segurança legítimas usadas por administradores de segurança de informações e testadores.

Como exemplo, "*John the Ripper*" era uma ferramenta de *cracking* de senhas de código aberto, usada originalmente por hackers para adivinhar senhas e acesso a redes ou aplicativos Unix. Hoje, esta ferramenta foi refinada e é usada para fins legítimos para detectar senhas fracas Unix. [Web-26]

Como principais fornecedores de ferramentas de teste e desenvolvimento de software e fornecedores de ferramentas especializadas começaram a desenvolver ferramentas de teste de segurança, muitas dessas ferramentas alcançaram capacidades funcionais mais amplas e usabilidade aprimorada. No entanto, essa ampla funcionalidade levou a configurações de ferramentas mais complexas e preocupações de implementação.

Ao mesmo tempo em que as primeiras ferramentas de segurança estavam surgindo, as primeiras versões de *frameworks* como *Nessus*, *Metasploit* e outros foram desenvolvidas como ferramentas de código aberto que oferecem funcionalidade melhorada e expandida e, em alguns casos, também uma GUI fácil de aprender.

Hoje, o número de ferramentas de teste de segurança disponíveis é enorme. Para quase qualquer ambiente ou tarefa, é possível encontrar uma ferramenta de teste dedicada, seja como código aberto ou licenciado. O desafio com todas essas ferramentas é que a maioria delas são sistemas inteligentes que implementam testes não padronizados. Todos os desenvolvedores desses sistemas concordam mais ou menos sobre como testar defesas de segurança ou testar vulnerabilidades. No entanto, essas ferramentas podem usar diferentes dados de teste, diferentes implementações de teste e diferentes interpretações dos resultados.

As ferramentas de teste de segurança podem ser usadas para automatizar a avaliação de defesas de segurança. Elas também podem ser usadas para detectar tipos conhecidos de vulnerabilidades. Com a compreensão de que o mesmo tipo de defesa ou vulnerabilidade de segurança pode ser implementado de diferentes maneiras, a seleção e o uso de ferramentas de teste de segurança é um desafio para o testador em segurança porque as ferramentas diferem em como encontrar vulnerabilidades e validar defesas.

Os Web sites do *Web Application Security Consortium* [Web-18] e *OWASP* [OWASP1] oferecem listas de ferramentas categorizadas. A estrutura de teste de penetração *Backtrack* [Web-23] (ou Kali Linux [Web-17]) apresenta outras formas de classificar as ferramentas de teste de segurança.

O número de ferramentas de segurança comercial é bastante limitado em comparação com o número de ferramentas de código aberto. Quando este syllabus foi desenvolvido (2016), conseguimos encontrar apenas um número limitado de recursos apresentando uma visão geral, mais ou menos completa, de ferramentas de segurança de código aberto confiáveis. Uma lista pode ser encontrada em https://sectools.org [Web-24]. Espera-se que o testador em segurança avançado mantenha sua própria lista de ferramentas disponíveis atualizando-a à medida que o mercado de ferramentas muda.

As ferramentas de análise estática e dinâmica são úteis em testes de segurança. A vantagem do teste estático é que ele pode ser realizado muito cedo no ciclo de vida do desenvolvimento. As ferramentas estáticas da análise estão disponíveis para a maioria das linguagens de desenvolvimento e têm geralmente uma habilidade de relatar os aspectos da segurança.

A diferença entre as ferramentas de teste dinâmico e estático no contexto do teste de segurança é às vezes pouco confusa em comparação com outros tipos de teste. A definição de teste estático está relacionada com a realização de atividades de teste enquanto o sistema ou objeto sob teste não está no modo operacional. Não é incomum para as ferramentas de teste dinâmico de segurança sondar o sistema em vez do aplicativo em teste. Nesta perspectiva, essas ferramentas são usadas como um tipo de ferramentas de teste estático. Por exemplo, uma ferramenta de teste de segurança dinâmica pode executar uma varredura estática de um banco de dados. Claro, se todo o sistema é considerado como o objeto de teste, então as ferramentas verdadeiramente são ferramentas de teste dinâmico.

8.2 Seleção da ferramenta

8.2.1 Analisando e documentando as necessidades de testes de segurança

Entre outros, os seguintes documentos podem formar uma base de teste para testes de segurança:

- Política de segurança da organização.
- Política de teste da organização.
- Resultados da análise de ameaças e riscos para o sistema / projeto atual.
- Requisitos e outras especificações do sistema.
- Arquitetura do sistema e modelagem.
- Estratégia de segurança (teste).
- O sistema ou aplicativo em teste.
- Alertas, vulnerabilidades e vulnerabilidades de segurança conhecidas.
- Perfis de usuário.

Todos estes e outros podem fornecer informações sobre ameaças e vulnerabilidades que poderiam ser exploradas. Os requisitos e documentos de projeto devem indicar como os dados ou informações estão protegidos. Isto levará a uma visão geral de:

- Interfaces a serem testadas (incluindo a GUI).
- Protocolos e padrões a serem verificados.
- Diretrizes de codificação da Web que promovem práticas seguras de codificação a serem utilizadas.
- Configurações dos componentes do sistema a serem verificados (endurecidos).

É preciso determinar se o teste de segurança será uma atividade de desenvolvimento ou uma atividade operacional. Todas essas informações levarão aos requisitos para o conjunto de ferramentas de teste de segurança.

8.2.2 Problemas com ferramentas de código aberto

Consulte [BSTQB_ATM_SYL] para obter uma discussão completa dos problemas que podem ser encontrados com as ferramentas de código aberto.

Como mencionado anteriormente, muitas ferramentas de teste de segurança são encontradas no domínio de código aberto. Essas ferramentas são distribuídas e podem ser usadas sob uma ampla variedade de licenças que permitem o uso e a modificação livre do código-fonte. Nem todas as empresas ou projetos podem considerar o uso de ferramentas de código aberto em seus processos de desenvolvimento. Com base em questões de conformidade regulamentar, as organizações podem ser forçadas a usar apenas ferramentas comerciais ou certificadas.

Existem muitas vantagens e desvantagens relacionadas a ferramentas sob essas licenças. Em muitos casos, as ferramentas de código aberto podem ser obtidas gratuitamente, mas a organização pode precisar ter capacidade técnica disponível para suporte e configuração específica. Se esta capacidade estiver faltando,

então um custo pode ser incorrido para obtê-lo a partir do desenvolvedor do software. Os manuais de administração e de usuário, se houver, são, em sua maioria, escritos por um público específico (técnico) em mente e, na maioria das vezes, não descrevem nem cobrem todas as funcionalidades da ferramenta. Os canais de mídia como o *YouTube* são recentemente uma fonte adicional de informações sobre o uso dessas ferramentas.

Aspectos a considerar ao configurar o cálculo ROI para qualquer ferramenta de código aberto (*opensource*) incluem:

- O escopo limitado da ferramenta (na maioria dos casos, nenhuma funcionalidade adicional ou outra é oferecida).
- O tempo para aprender a administrar, configurar e usar a ferramenta.
- O tempo para investir nos fóruns e grupos de usuários durante o ciclo de vida.
- O tempo necessário para atualizar (e a política interna sobre atualizações).
- A direção futura da ferramenta (algumas ferramentas podem desaparecer ou tornarem-se comerciais).
- O nível de resposta na comunidade de suporte para a ferramenta.

Para a maioria das empresas ou projetos, o número de licenças necessárias para as ferramentas de teste de segurança é limitado. Apenas as grandes empresas considerarão mais licenças. O número de licenças basear-se-á principalmente na soma total das áreas de funcionalidade fornecidas pela ferramenta (p.e., aplicação web, serviços web, análise de código, outros) e a frequência assumida, o tempo de utilização destes serviços e o número de testadores em segurança utilizando a ferramenta.

8.2.3 Avaliando as capacidades do vendedor de uma ferramenta

Se uma ferramenta é comprada de um fornecedor, esse deve oferecer uma série de serviços para permitir que a atividade de teste de segurança se adeque para o nível necessário de suporte interno.

Os seguintes atributos podem ser usados para avaliar os recursos do fornecedor:

- Tipos de licenças oferecidas (fixo / *desk* / *floating* / *token*).
- Opções de escalabilidade de licenças (por área funcional, número de licenças).
- *Helpdesk* e instalações de apoio (horas de apoio).
- Fóruns e comunidades de usuários.
- Frequência de atualização.
- Manuais de administração e de usuário.
- Contratos de suporte e manutenção .

9 Padrões e tendências da indústria (40 min)

Palavras-chave

consensus-based standard

Objetivos de aprendizagem

9.1 Compreendendo os padrões de testes de segurança

9.1.1 (K2): Compreender os benefícios do uso de padrões de testes de segurança e onde encontrá-los.

9.1.2 (K2): Compreender a diferença na aplicabilidade das normas em situações regulatórias versus contratuais.

9.2 Aplicação de padrões de segurança

9.2.1 (K2): Compreender a diferença entre as cláusulas obrigatória (normativa) e opcional (informativa) dentro de qualquer norma.

9.3 Tendências da indústria

9.3.1 (K2): Entender onde aprender das tendências da indústria em matéria de segurança da informação.

9.1 Compreendendo os padrões de testes de segurança

Padrões de vários tipos fornecem visibilidade sobre o consenso profissional ou obrigações regulatórias. Um padrão consensual representa o parecer de um grupo de peritos bem informado e é disponibilizado para uso voluntário (total ou parcial) em acordos contratuais entre fornecedores e clientes. Existem tipos menores de padrões que surgem de grupos mais informais ou auto-identificados e podem ser específicos a alguns fornecedores.

Nas indústrias regulamentadas (incluindo os setores médico, financeiro, de transporte e de energia), as agências governamentais podem exigir o cumprimento de seus próprios regulamentos ou suas interpretações de padrões voluntários.

9.1.1 Os benefícios do uso de padrões de testes de segurança

Padrões, em geral, fornecem orientação e consistência na execução de uma tarefa. Normalmente, os padrões são desenvolvidos por especialistas em matéria de assunto com base no consenso de práticas eficazes. Os seguintes são benefícios de usar padrões de teste de segurança:

- Eles definem uma estrutura para testes de segurança eliminando a necessidade de iniciar a partir de uma página em branco.
- Eles descrevem defesas eficazes e como testar os ataques de segurança mais comuns.
- Os padrões podem ser adaptados para atender às necessidades do projeto ou da organização.
- A devida diligência nos testes de segurança pode ser demonstrada seguindo padrões de teste de segurança reconhecidos.

9.1.2 Aplicabilidade de normas em situações regulamentares versus contratuais

Em atividades regulamentadas, todas as partes precisam estar cientes de suas obrigações de cumprir as normas impostas, pois o não cumprimento dessa medida pode retardar ou impedir a aprovação do produto em desenvolvimento e, em casos extremos, resultar em sanções financeiras ou criminais.

Em situações contratuais, as normas fornecem uma base razoável e conveniente para negociar o acordo sobre os requisitos do projeto e do produto. Eles fornecem um ponto de partida em vez das partes que começam com nada. As normas baseadas no consenso permitem que as melhores práticas sejam comunicadas e adotadas ou adaptadas à situações específicas.

A menos que seja imposta unilateralmente por um regulador ou em um contrato não negociável, as normas podem ser usadas como o quadro básico para um acordo negociado ou auto-imposto sobre a conduta do próprio trabalho. Se um contrato é adjudicado com base em uma reivindicação ou acordo para cumprir com normas específicas, então a entidade tem a obrigação de seguir essas normas estritamente e documentar quaisquer desvios.

9.1.3 Seleção de padrões de segurança

Certamente, nem todos os padrões de segurança se aplicam a todas as situações. É da responsabilidade de uma organização pesquisar os padrões mais apropriados para seus sistemas, aplicações, ativos digitais sigilosas, nível de risco e requisitos de conformidade. Também é importante entender que muitos padrões podem ser adaptados para atender às necessidades específicas de uma organização.

Uma lista de normas de segurança comuns pode ser encontrada no Capítulo 10.

9.2 Aplicação de padrões de segurança

Observe o uso preciso da linguagem dentro de qualquer padrão: a palavra deve identificar os requisitos obrigatórios a serem seguidos de acordo com a norma, enquanto as palavras devem e podem indicar tarefas opcionais que não são obrigados a reivindicar a conformidade com a norma. Um mau uso típico é confundir esta distinção quer exigindo um item opcional ou tratando um item obrigatório como opcional.

Situações específicas da organização ou do projeto podem ditar o desvio do senso estrito de um padrão em uso. Justificação para omissões, modificações ou adições ao conteúdo da norma precisam ser documentadas e acordadas por todas as partes.

9.3 Tendências da indústria

9.2.1 Onde aprender das tendências da indústria na segurança da informação

Tanto os serviços de notícias de uso geral e específicos da indústria (publicações, websites, distribuições de e-mail) e eventos (conferências, feiras comerciais, reuniões da sociedade profissional) apresentam informações e discussão de novas ou crescentes preocupações. Pertencer a uma sociedade ou comunidade profissional focada provavelmente fornecerá atualizações oportunas e direcionadas. Com a velocidade com que novas façanhas se desenvolvem, os alertas eletrônicos podem oferecer as respostas mais imediatas.

A divulgação periódica das explorações mais frequentes ou prejudiciais pode identificar tendências generalizadas, mas deve-se prestar especial atenção a questões mais específicas para a indústria, área de aplicação ou produtos com os quais se está trabalhando. Estas questões são mais susceptíveis de serem comunicadas em publicações especializadas e serviços de notícias ou em conferências técnicas e eventos profissionais.

9.2.2 Avaliando práticas de testes de segurança para melhorias.

À medida que novas tecnologias ou novos usos da tecnologia existentes são introduzidos, muitas vezes há uma janela de oportunidade para uso indevido ou exploração da tecnologia até que seus riscos e limitações sejam mais bem compreendidos.

Por exemplo, considere dispositivos móveis com serviços de reconhecimento de local. Em troca de conveniência ou outros incitamentos, os indivíduos parecem dispostos a permitir o rastreamento minuto a minuto de seus movimentos e atividades.

Uma gama maior de motivações e maiores recursos estão emergindo de agentes criminais, *hacktivistas*, econômicos e políticos. Os esquemas de extorsão e proteção passaram de ameaças físicas para domínios digitais.

Grandes redes *ad hoc* de indivíduos ideologicamente orientados podem ser direcionadas em muito curto prazo contra alvos de sua ira. Espionagem corporativa é muitas vezes bem financiado e motivado. As Nações-Estado que buscam vantagens econômicas e militares são particularmente bem-dotadas de recursos e podem acreditar que estão imunes a sanções ou retaliações.

Como as ameaças estão constantemente mudando e evoluindo, os testadores em segurança devem estar sempre prontos para enfrentar a próxima ameaça. A conscientização da indústria, o acompanhamento rigoroso das tendências de segurança e a aquisição das ferramentas mais adequadas proporcionam a melhor defesa para uma organização.

10 Referencias

10.1 Documentos BSTQB

- [BSTQB_FL_SYL] BSTQB Foundation Syllabus, 2011
- [BSTQB_ATM_SYL] BSTQB Advanced Test Manager Syllabus, 2012
- [BSTQB_ATTA_SYL] BSTQB Advanced Technical Test Analyst Syllabus, 2012

10.2 Normas e padrões

- [ISO/IEC/IEEE 29119-3] Software and Systems Engineering, Software testing, Part 3: Test documentation
- [IEEE 12207] Standard for Systems and Software Engineering, Software Life Cycle Processes
- [COBIT] www.isaca.org
- [ISO27001] Information Security Management, www.iso.org/iso/home/standards/managementstandards/iso27001.htm
- [PCI] Payment Card Industry Standard, www.pcisecuritystandards.org/

10.3 Literatura

- [Chapman, 2000] Chapman, Cooper, Zwicky, Building Internet Firewalls, O'Reilly & Associates, 2000.
- [Jackson, 2010] Jackson, Christopher; Network Security Auditing, 2010.

10.4 Artigos

- [ComputerWeekly] www.computerweekly.com/news/2240113549/Cattles-lost-backup-tapeshighlight-risk-of-unencrypted-data-storage
- [Northcutt, 2014] Northcutt, Stephen; Security Controls, SANS Institute.
- [Washington Post, 2007], www.washingtonpost.com/wpdyn/content/article/2007/05/04/AR2007050402152.html

10.5 Guias

- [Bittau] Cryptographic Protection of TCP Streams, tools.ietf.org/html/draft-bittau-tcp-crypt-04
- [CERT1] Top 10 Secure Coding Practices, www.securecoding.cert.org/confluence/display/seccode/Top+10+Secure+Coding+Practices
- [CERT2] www.cert.org/secure-coding/publications/index.cfm
- [CERT3] www.cert.org/secure-coding/tools/index.cfm
- [IEEE1] Avoiding the Top 10 Security Flaws, cybersecurity.ieee.org/center-for-secure-modelagem/avoiding-the-top-10-security-flaws.html
- [MDA1] MDA Glossary, DoD Missile Defense Agency, www.mda.mil
- [NIST 800-30] NIST Special Publication 800-30, Rev 1, Guide for Conducting Risk Assessments (2012)
- [NISTIR 7298] Glossary of Key Information - Security Terms, Revision 2 (2013)
- [OWASP1] OWASP Secure Coding Practices Quick Reference Guide, www.owasp.org/index.php/OWASP_Secure_Coding_Practices_-_Quick_Reference_Guide
- [OWASP2] OWASP Risk Rating Methodology, www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

- [OWASP3] OWASP Sample Authorization Form, www.owasp.org/index.php?title=Authorization_form
- [PP-IDS] US Government Protection Profile Intrustion Detection System for basic robustness environments, version 1.7, 25 July 2007.
- [SANS1] 25 Most Dangerous Software Errors, www.sans.org
- [SANS2] Password Construction Guidelines, www.sans.org/securityresources/policies/general/pdf/password-construction-guidelines
- [WAFEC] Web Application Firewall Evaluation Criteria, wasc-wafec-v1.0.pdf, 2006.

10.6 Relatórios

- [WhiteHat Security, 2014], www.whitehatsec.com

10.7 Web

- [CERT4] Vulnerability Notes Database, www.kb.cert.org/vuls/
- [Chopitea] tomchop.me/2012/12/yo-dawg-i-heard-you-like-xoring/
- [EICAR] www.eicar.org
- [RFC2828] Internet Security Glossary, www.rfc-archive.org/getrfc.php?rfc=2828
- [Web-1] Top 20 Critical Security Controls, sans.org
- [Web-2] National Vulnerability Database, web.nvd.nist.gov/view/ncp/repository
- [Web-3] Website Security Statistics Report, www.whitehatsec.com/resource/stats.html
- [Web-4] The Google Hacking Database, hackersforcharity.org/ghdb
- [Web-5] Shodan, shodanhq.com
- [Web-6] NetCat, sectools.org/tool/netcat/
- [Web-7] IP Tracer, www.ip-adress.com/ip_tracer
- [Web-8] Computer Forensics, Cybercrime and Steganography Resources, www.forensics.nl
- [Web-9] Tor Project, www.torproject.org/
- [Web-10] Netstat, technet.microsoft.com/en-us/library/Bb490947.aspx
- [Web-11] Tracert, www.tracert.com
- [Web-12] RIPE Scan, www.ripe.net
- [Web-13] Whois, www.whois.net/
- [Web-14] NetCat, netcat.sourceforge.net/
- [Web-15] Fping, fping.org
- [Web-16] Hidetools, hidetools.com/
- [Web-17] Kali Linux, www.kali.org/
- [Web-18] Web Application Security Consortium, www.webappsec.org/
- [Web-19] Hping, www.hping.org/
- [Web-20] Nmap, nmap.org/
- [Web-21] Zenmap, nmap.org/zenmap/
- [Web-22] Xprobe2, null-byte.wonderhowto.com/how-to/hack-like-pro-conduct-os-fingerprintingwith-xprobe2-0148439/
- [Web-23] BackTrack, www.backtrack-linux.org/
- [Web-24] Top 125 Network Security Tools, sectools.org
- [Web-25] DNS Lookup, who.is/dns/
- [Web-26] John the Ripper, www.openwall.com/john/