Pentest Web

Introdução:

O pentest web é um teste de invasão ou de segurança que se concentra apenas em aplicação web, o objeto e encontrar vulnerabilidades nos sites ou sistemas web.

Ferramentas são usadas para encontrar vetores de ataques muitas dessas ferramentas são automatizadas para facilitar no processo de pentest.

Podemos citar as seguintes ferramentas Burp suit, SqLMaP, Hidra, OWAP Zap entre outras.

Fundamentos:

A fundação OWASP mantém uma lista atualizada das 10 vulnerabilidades mais comuns encontradas em aplicações web

Organizadas da seguinte forma:

A1: 2017-Injection: Falhas de injeção, como SQL, NoSQL, OS e injeção de LDAP, ocorrem quando dados não confiáveis são enviados a um intérprete como parte de um comando ou consulta. Os dados hostis do invasor podem induzir o intérprete a executar comandos indesejados ou acessar dados sem a autorização adequada.

A2: Autenticação quebrada em 2017: as funções do aplicativo relacionadas à autenticação e gerenciamento de sessão são frequentemente implementadas incorretamente, permitindo que os invasores comprometam senhas, chaves ou tokens de sessão ou explorem outras falhas de implementação para assumir as identidades de outros usuários temporária ou permanentemente.

A3: Exposição de dados confidenciais de 2017: Muitos aplicativos da web e APIs não protegem adequadamente os dados confidenciais, como finanças, saúde e PII. Os invasores podem roubar ou modificar esses dados fracamente protegidos para conduzir fraude de cartão de crédito, roubo de identidade ou outros crimes. Os dados confidenciais podem ser comprometidos sem proteção extra, como criptografia em repouso ou em trânsito, e requerem precauções especiais quando trocados com o navegador.

A4: 2017-XML Entidades externas (XXE): Muitos processadores XML mais antigos ou mal configurados avaliam referências de entidades externas em documentos XML. Entidades externas podem ser usadas para divulgar arquivos internos usando o manipulador de URI de arquivo, compartilhamentos de arquivos internos, varredura de porta interna, execução remota de código e ataques de negação de serviço.

A5: 2017-Broken Access Control: as restrições sobre o que os usuários autenticados têm permissão para fazer muitas vezes não são aplicadas de forma adequada. Os invasores podem explorar essas falhas para acessar funcionalidades e / ou dados não autorizados, como acessar contas de outros usuários, visualizar arquivos confidenciais, modificar dados de outros usuários, alterar direitos de acesso, etc.

A6: 2017-Configuração incorreta de segurança: a configuração incorreta de segurança é o problema mais comum. Isso geralmente é o resultado de configurações padrão inseguro, configuração incompleta ou ad hoc, armazenamento em nuvem aberta, cabeçalhos HTTP configurados incorretamente e mensagens de erro detalhadas contendo informações

confidenciais. Não apenas todos os sistemas operacionais, estruturas, bibliotecas e aplicativos devem ser configurados com segurança, mas também devem ser corrigidos / atualizados em tempo hábil.

A7: 2017-Cross-Site Scripting XSS: As falhas de XSS ocorrem sempre que um aplicativo inclui dados não confiáveis em uma nova página da web sem validação ou escape adequado, ou atualiza uma página da web existente com dados fornecidos pelo usuário usando uma API do navegador que pode criar HTML ou JavaScript. O XSS permite que os invasores executem scripts no navegador da vítima, que podem sequestrar as sessões do usuário, desfigurar sites ou redirecionar o usuário para sites maliciosos.

A8: 2017-Desserialização insegura: a desserialização insegura geralmente leva à execução remota de código. Mesmo que as falhas de desserialização não resultem na execução remota de código, elas podem ser usadas para realizar ataques, incluindo ataques de repetição, ataques de injeção e ataques de escalonamento de privilégios.

A9: 2017 - Usando componentes com vulnerabilidades conhecidas: componentes, como bibliotecas, estruturas e outros módulos de software, são executados com os mesmos privilégios do aplicativo. Se um componente vulnerável for explorado, esse tipo de ataque pode facilitar a perda séria de dados ou o controle do servidor. Aplicativos e APIs que usam componentes com vulnerabilidades conhecidas podem minar as defesas do aplicativo e permitir vários ataques e impactos.

A10: Registro e monitoramento insuficientes em 2017: registro e monitoramento insuficientes, juntamente com a integração

ausente ou ineficaz com a resposta a incidentes, permite que os invasores ataquem ainda mais os sistemas, mantenham a persistência, pivotem para mais sistemas e adulterem, extraiam ou destruam dados. A maioria dos estudos de violação mostra que o tempo para detectar uma violação é de mais de 200 dias, normalmente detectado por partes externas em vez de processos internos ou monitoramento.

Desenvolvimento:

O pentester tem seu principal foco as essas 10 vulnerabilidades, mas deve segue sempre o escopo definido no acordo entre as partes, aqui são definidos o objetivo do pentest ou auditoria, são incluídas as condições e restrições do cliente.

Um pentest segue as seguintes etapas:

Metodologia baseada em cinco passos:

- (1) – Reconhecimento:

Esta fase tem como foco aprender tudo sobre o alvo. Documentando o máximo possível de informações sobre o alvo.

Espelhamento de sites, Pesquisas no Google, Google hacking, Mídias sociais, Sites de ofertas de emprego, DNS e ataques DNS

Principais ferramentas: Netcraft, Whois, Nslookup, Dig e Engenharia social

- (2) Scanning:

Scanning de portas e scanning de vulnerabilidades. Possibilita ter uma melhor definição da rede e da infraestrutura do sistema de informação que será alvo da exploração de falhas

Principais técnicas: Conhecimentos de protocolos (TCP, UDP, IP e ICMP)

Principais ferramentas: Nmap, Zenmap, Hping3 e Nessus.

- **(3) Exploração de falhas:**

Nesta faze após obter as informações inicia-se o ataque aos alvos através de várias técnicas e ferramentas. Objetivo final é ter acesso completo (administrador) sobre o alvo. Pode ser local ou remota. Entrar no sistema alvo e sair com informações sem ser notado usando as vulnerabilidades e técnicas comprovadas.

Principais técnicas: Teste de vulnerabilidades, Quebra de senhas Obter acesso a serviços e Sniffing do tráfego de rede.

Principais ferramentas: Nikto, Metasploit, Burp Suite, Zed Attack Proxy (ZAP) e John the Ripper (JtR).

- (4) Preservação do acesso:

Cria-se uma porta dos fundos permanente para acesso ao sistema. Após explorar falhas, backdoors e rootkits serão deixados nos sistemas para permitir acesso futuro.

Principais técnicas: Instalação de backdoors e Instalação de rootkits

Principais ferramentas: Netbus, Sub7 e Back orifice (Backdoors) , Netcat (canivete suíço), Hacker defender (rootkit) e Meterpreter (payload).

- (5) Gerando relatórios: Gerar relatórios detalhados gerenciais e técnicos sobre o alvo para explicar o processo de hacking.

Ações do relatório:

- ✓ Sumário executivo
- ✓ Procedimentos ligados ao teste
- ✓ Arquitetura e composição do alvo
- ✓ Descobertas
- ✓ Ações recomendadas

Elaboração do relatório:

- ✓ Conclusões
- ✓ Apêndices
- ✓ Apresentação
- ✓ Armazenamento do relatório e evidências

Conclusão:

O pentest Web é uma forma de define, e ou encontrar vulnerabilidades em aplicações web ou sites a fim de sanar ocorrências de acesso ou abuso ao sistema de forma de indevida diminuindo assim a vasão de dados sensíveis, é também uma forma de demonstrar à adequação a nova lei de proteção de dados que visa evitar o vazamento e abuso em relação aos dados.

Referências:

https://www.infosec.com.br/pentest-web/ - Pentest Web.

PAULI, Josh – Introdução ao hacking e aos testes de invasão: Facilitando o hacking ético e os testes de invasão – São Paulo: Novatec, 2014.

https://owasp.org - The Open Web Application Security Project