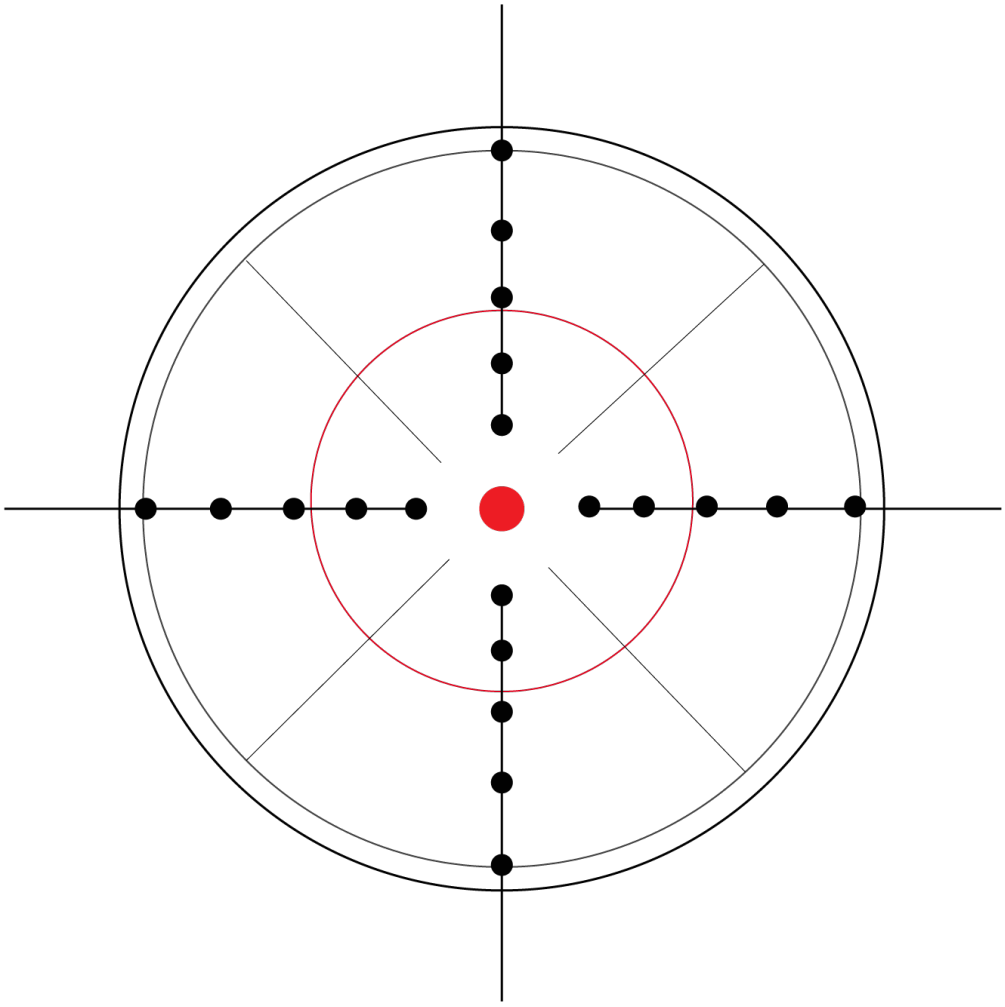
O que é SAST e DAST?



Junho/2024

mindsec.com.br

Diagramado e escrito por: Mindsec Segurança e Tecnologia

CONTEÚDO

O que é SAST e DAST?

SAST e DAST são duas abordagens distintas para testes de segurança de software, cada uma focando em aspectos específicos do processo de desenvolvimento e segurança de aplicações. Vamos detalhar cada uma delas:

SAST (Static Application Security Testing)

SAST é uma técnica de teste de segurança que analisa o código-fonte ou o código compilado de um aplicativo em busca de vulnerabilidades de segurança, sem executar o programa.

Utiliza análise estática para identificar potenciais falhas de segurança, como vulnerabilidades de SQL injection, cross-site scripting (XSS), uso inseguro de funções de criptografia, entre outros.

Pode ser integrado no ciclo de desenvolvimento de software (SDLC) para identificar problemas de segurança desde as fases iniciais de desenvolvimento até o lançamento.

Funcionamento

Examina o código-fonte ou o código compilado sem executar o aplicativo.

Identifica padrões de código que podem indicar vulnerabilidades conhecidas.

Gera relatórios detalhados com recomendações de correções para os desenvolvedores.

- **DAST (Dynamic Application Security Testing):**

DAST é uma abordagem de teste de segurança que avalia a segurança de uma aplicação enquanto ela está em execução.

Simula ataques de um ponto de vista externo, interagindo com o aplicativo como faria um usuário real.

Identifica vulnerabilidades que podem ser exploradas através de ataques de injeção de SQL, XSS, e falhas de configuração de segurança.

Funcionamento

Realiza varreduras automatizadas ou controladas manualmente para simular ataques de um usuário externo.

Identifica vulnerabilidades exploráveis que não são visíveis na análise estática.

Gera relatórios detalhados com informações sobre as vulnerabilidades encontradas e possíveis soluções.

Comparação SAST vs DAST

- **SAST** é aplicável desde as fases iniciais de desenvolvimento, identificando problemas de segurança no código-fonte ou código compilado.
- **DAST**, por outro lado, verifica a segurança do aplicativo em tempo de execução, simulando ataques reais e identificando vulnerabilidades que podem ser exploradas externamente.

Ambas as abordagens são complementares e desempenham papéis importantes na garantia da segurança de software. Idealmente, as organizações utilizam ambas as técnicas para cobrir tanto as vulnerabilidades detectáveis estaticamente quanto as que só podem ser identificadas durante a execução do aplicativo.

SAST (STATIC APPLICATION SECURITY TESTING):

1.1. Metodologia

O SAST realiza uma análise estática do código-fonte ou do código compilado de um aplicativo. Isso significa que examina o código sem executar o programa, focando na estrutura do código, fluxo de dados e chamadas de função.

1.2. Funcionamento Detalhado do SAST (Static Application Security Testing)

1.2.1. Preparação e Configuração

Antes de iniciar a análise, é necessário configurar a ferramenta de SAST para o ambiente de desenvolvimento específico da organização. Isso pode incluir a definição de quais linguagens de programação e frameworks serão analisados, além de ajustes nas configurações de segurança e regras de análise.

1.2.2. Coleta de Código-Fonte

O SAST realiza uma análise estática do código-fonte ou do código compilado do aplicativo. Essa análise examina o código sem executar o programa, focando na estrutura do código, fluxo de dados e chamadas de função.

1.2.3. Processo de Análise

- **Identificação de Vulnerabilidades Conhecidas**: A ferramenta de SAST utiliza um banco de dados de vulnerabilidades conhecidas e padrões de código inseguros para identificar potenciais falhas de segurança. Isso inclui vulnerabilidades comuns como injeção de SQL, XSS (Cross-Site Scripting), deserialização inadequada, uso inseguro de APIs, entre outros.
- **Análise de Fluxo de Dados**: Verifica como os dados são manipulados e passados dentro do código, procurando por vulnerabilidades como vazamento de informações sensíveis.
- **Análise de Controle de Fluxo**: Examina as condições e ramificações do código para verificar se há pontos fracos que poderiam ser explorados por atacantes.
- **Análise de Criptografia**: Verifica se os algoritmos de criptografia são implementados de maneira segura e se as chaves são gerenciadas corretamente.

1.2.4. Geração de Relatórios e Resultados

Após concluir a análise, o SAST gera relatórios detalhados que listam as vulnerabilidades encontradas, juntamente com informações sobre a localização no código, o risco associado e recomendações de correção.

Algumas ferramentas de SAST são capazes de priorizar as vulnerabilidades com base na gravidade, impacto potencial e facilidade de correção, permitindo que os desenvolvedores foquem nos problemas mais críticos primeiro.

1.2.5. Integração com o Ciclo de Desenvolvimento

O SAST pode ser integrado diretamente ao ciclo de vida de desenvolvimento de software (SDLC), fornecendo feedback contínuo aos desenvolvedores durante todo o processo de desenvolvimento.

Muitas organizações automatizam a execução de testes de SAST como parte de pipelines de integração contínua (CI) e entrega contínua (CD), garantindo que novos códigos sejam verificados quanto a vulnerabilidades antes de serem implantados em produção.

1.3. Benefícios do SAST (Static Application Security Testing)

1.3.1. Detecção Precoce de Vulnerabilidades

O SAST identifica vulnerabilidades de segurança desde as fases iniciais do ciclo de desenvolvimento de software, analisando o código-fonte ou código compilado antes mesmo da execução do programa.

Isso permite que os desenvolvedores corrijam as vulnerabilidades antes que o software seja implantado em ambientes de produção, reduzindo significativamente os custos e o tempo associados à correção de problemas descobertos tardiamente.

1.3.2. Integração com o Ciclo de Desenvolvimento (SDLC)

Integrado ao SDLC, o SAST fornece feedback contínuo aos desenvolvedores durante todo o processo de desenvolvimento.

Isso promove uma cultura de segurança desde o início do desenvolvimento, garantindo que as melhores práticas de segurança sejam incorporadas ao código desde o momento da sua criação.

1.3.3. Conformidade com Padrões e Regulamentações

Muitas organizações são obrigadas a cumprir regulamentações e padrões de segurança, como PCI-DSS, GDPR, HIPAA, entre outros.

O SAST ajuda as organizações a identificar e corrigir vulnerabilidades que poderiam levar a violações de conformidade, mitigando riscos legais e financeiros associados a violações de dados.

1.3.4. Identificação de Vulnerabilidades Complexas

Examina o código-fonte em busca de vulnerabilidades complexas que podem não ser facilmente detectadas por ferramentas de teste de segurança dinâmica (DAST).

Isso inclui vulnerabilidades como manipulação inadequada de dados, vazamento de informações sensíveis, falhas de autenticação e autorização, entre outros, que podem ser críticas para a segurança do aplicativo.

1.3.5. Redução de Riscos de Segurança

Corrigir vulnerabilidades identificadas pelo SAST ajuda a mitigar riscos de segurança antes que possam ser explorados por hackers ou atacantes.

Isso fortalece a postura de segurança da organização, protegendo ativos críticos e reduzindo a superfície de ataque disponível para potenciais ameaças.

1.3.6. Melhoria da Qualidade do Código

O SAST promove o uso de práticas de codificação seguras, como uso correto de APIs, gerenciamento adequado de memória, e tratamento de exceções.

Além de melhorar a segurança, essas práticas também contribuem para a qualidade geral do código, reduzindo a ocorrência de bugs e aumentando a eficiência operacional do software.

1.3.7. Suporte à Automação e Integração Contínua (CI/CD)

O SAST pode ser integrado a pipelines de integração contínua (CI) e entrega contínua (CD), automatizando a execução de testes de segurança em cada alteração de código.

Isso garante que novos códigos sejam verificados quanto a vulnerabilidades de segurança antes de serem implantados em produção, mantendo um alto nível de segurança sem comprometer a velocidade de desenvolvimento.

O SAST oferece uma abordagem fundamental para identificar e mitigar vulnerabilidades de segurança no código-fonte de aplicações, proporcionando benefícios significativos que vão desde a detecção precoce de problemas até a conformidade com regulamentações e melhoria da qualidade do código. Integrar o SAST no ciclo de vida de desenvolvimento de software é essencial para fortalecer a postura de segurança cibernética de uma organização e proteger seus ativos críticos contra ameaças em constante evolução.

1.4. Exemplos de Análises Realizadas pelo SAST

O SAST utiliza uma variedade de técnicas de análise estática para identificar potenciais vulnerabilidades de segurança no código-fonte ou no código

compilado de um aplicativo. Aqui estão alguns exemplos detalhados das análises realizadas:

1.4.1. Análise de Fluxo de Dados

- **Objetivo**: Verifica como os dados são manipulados e passados dentro do código.
- **Exemplo**: Identificação de pontos onde dados sensíveis são armazenados sem criptografia adequada ou onde podem ser acessados sem a devida autenticação.

1.4.2. Análise de Controle de Fluxo

- **Objetivo**: Examina as condições e ramificações do código para identificar possíveis vulnerabilidades de segurança.
- **Exemplo**: Detecção de código que permite acesso não autorizado a recursos ou funcionalidades sensíveis através de condições mal implementadas ou falta de validação adequada.

1.4.3. Análise de Manipulação de Entrada de Usuário

- **Objetivo**: Verifica como o código trata entradas de usuário potencialmente maliciosas.
- **Exemplo**: Identificação de vulnerabilidades de injeção de SQL, onde entradas não validadas de usuários podem ser usadas para manipular bancos de dados ou obter informações sensíveis.

1.4.4. Análise de APIs e Frameworks

- **Objetivo**: Avalia como APIs e frameworks são utilizados no código e se são implementados de maneira segura.
- **Exemplo**: Identificação de uso inadequado de APIs que podem expor dados sensíveis, como tokens de autenticação ou chaves de API, a potenciais ataques.

1.4.5. Análise de Criptografia

- **Objetivo**: Verifica se os algoritmos de criptografia são implementados de maneira segura e se as chaves são gerenciadas adequadamente.
- **Exemplo**: Identificação de uso de algoritmos fracos de criptografia, chaves hardcoded no código-fonte, ou falta de renovação regular de chaves criptográficas.

1.4.6. Análise de Conformidade com Padrões de Codificação

- **Objetivo**: Avalia se o código está em conformidade com padrões de segurança e boas práticas de codificação.
- **Exemplo**: Identificação de padrões de codificação inseguros, como uso de funções depreciadas, uso inadequado de variáveis não inicializadas, ou falta de tratamento de exceções.

1.4.7. Análise de Configurações de Segurança

- **Objetivo**: Verifica se as configurações de segurança do ambiente de desenvolvimento são adequadas.

- **Exemplo**: Identificação de configurações incorretas que podem expor o aplicativo a ataques, como permissões excessivas em arquivos ou serviços não essenciais habilitados.

Esses exemplos ilustram como o SAST é uma ferramenta essencial para mitigar riscos de segurança ao identificar e corrigir vulnerabilidades potenciais antes que possam ser exploradas por atacantes. Integrar o SAST no processo de desenvolvimento de software é fundamental para fortalecer a segurança cibernética de uma organização e proteger seus ativos críticos contra ameaças.

1.5. Exemplos de Ferramentas SAST

Veracode:

- **Descrição:** Plataforma de análise estática avançada que identifica vulnerabilidades de segurança no código-fonte, proporcionando análises detalhadas e relatórios de correção.
- **Recursos:** Integração com CI/CD, suporte a várias linguagens de programação e escalabilidade para grandes projetos.

Checkmarx:

- **Descrição**: Oferece análise estática de código para identificar vulnerabilidades desde as fases iniciais de desenvolvimento.
- **Recursos**: Verificação abrangente de segurança com suporte a linguagens de programação populares, relatórios detalhados e integração com ferramentas de desenvolvimento.

Fortify (Micro Focus Fortify):

- **Descrição**: Combina análise estática e dinâmica para identificar e mitigar vulnerabilidades de segurança em código-fonte e aplicações.

- **Recursos**: Análise profunda de código, suporte a várias linguagens, gerenciamento centralizado de políticas de segurança e integração com IDEs.

SonarQube:

- **Descrição**: Ferramenta de código aberto para inspeção contínua da qualidade do código, incluindo análise estática para segurança.
- **Recursos**: Verificações automáticas de código, detecção de vulnerabilidades de segurança e conformidade com padrões de codificação.

IBM Security AppScan:

- **Descrição**: Plataforma de segurança que oferece análise estática e dinâmica para identificar vulnerabilidades em aplicações web e móveis.
- **Recursos**: Escaneamento automatizado, relatórios detalhados de vulnerabilidades, integração com ferramentas de desenvolvimento e suporte a várias linguagens.

WhiteHat Sentinel:

- **Descrição**: Plataforma de segurança baseada em nuvem que oferece testes estáticos e dinâmicos para identificar vulnerabilidades em aplicações web.
- **Recursos**: Escaneamento contínuo, relatórios de segurança acionáveis, e suporte à integração com ferramentas de desenvolvimento.

CAST Application Intelligence Platform (AIP):

- **Descrição**: Plataforma que combina análise estática e dinâmica para avaliar a segurança, qualidade e resiliência do software.

- **Recursos**: Análise detalhada de código, monitoramento contínuo de segurança e suporte a várias linguagens de programação.

Klocwork (Rogue Wave Software):

- **Descrição**: Ferramenta de análise estática que identifica vulnerabilidades de segurança e problemas de qualidade no código-fonte.
- **Recursos**: Verificação automática de código, identificação de padrões de segurança, e integração com IDEs populares.

AppSpider (Rapid7):

- **Descrição**: Oferece análise estática e dinâmica para identificar vulnerabilidades em aplicações web e APIs.
- **Recursos**: Escaneamento automatizado, relatórios detalhados de segurança, e suporte à integração com pipelines de CI/CD.

Qualys Web Application Scanning (WAS):

- **Descrição**: Plataforma que combina testes de segurança estáticos e dinâmicos para identificar e corrigir vulnerabilidades em aplicações web.
- **Recursos**: Escaneamento contínuo, relatórios de conformidade e integração com outras soluções de segurança da Qualys.

1.6. Benefícios das Ferramentas de SAST

- **Identificação Precisa de Vulnerabilidades**: Ferramentas de SAST oferecem uma análise detalhada que identifica potenciais falhas de segurança no código-fonte ou no código compilado.

- **Integração com o Ciclo de Desenvolvimento**: Facilitam a integração contínua no ciclo de desenvolvimento de software, proporcionando feedback imediato aos desenvolvedores.
- **Melhoria Contínua da Segurança**: Promovem práticas de desenvolvimento seguro e melhoram a postura de segurança cibernética da organização.

Essas ferramentas são essenciais para organizações que buscam fortalecer a segurança de suas aplicações, garantindo que vulnerabilidades sejam identificadas e corrigidas antes de serem exploradas por atacantes.

1.7. Conclusão sobre Ferramentas de SAST

As ferramentas de SAST desempenham um papel crucial na identificação e mitigação de vulnerabilidades de segurança no código-fonte das aplicações. Ao oferecer uma análise detalhada e automatizada, essas ferramentas ajudam as organizações a fortalecer sua postura de segurança cibernética desde as fases iniciais do desenvolvimento de software. Os benefícios incluem:

- **Identificação Precisa de Vulnerabilidades**: As ferramentas de SAST destacam vulnerabilidades de injeção de SQL, XSS, problemas de criptografia, entre outros, proporcionando insights detalhados sobre sua localização e risco potencial.
- **Integração com o Ciclo de Desenvolvimento**: Integradas aos ambientes de CI/CD e IDEs, as ferramentas de SAST oferecem feedback contínuo aos desenvolvedores, permitindo correções rápidas e eficazes durante o processo de desenvolvimento.
- **Melhoria Contínua da Segurança**: Ao promover práticas de codificação seguras e identificar vulnerabilidades antes da

implantação, as ferramentas de SAST ajudam a mitigar riscos e fortalecer a segurança cibernética da organização.

- **Suporte a Diversas Linguagens e Frameworks**: Com suporte abrangente a várias linguagens de programação e tecnologias, as ferramentas de SAST são adaptáveis a diferentes ambientes de desenvolvimento, garantindo análises precisas e relevantes.

Em resumo, investir em ferramentas de SAST não apenas ajuda a proteger aplicações contra ameaças cibernéticas, mas também contribui para a conformidade com regulamentações de segurança e a melhoria contínua da qualidade do software. Ao incorporar análises estáticas robustas no processo de desenvolvimento, as organizações podem garantir que suas aplicações sejam seguras, confiáveis e resilientes contra ataques em um ambiente digital cada vez mais complexo e desafiador.

DAST (Dynamic Application Security Testing)

1.8. Metodologia

DAST, ou Dynamic Application Security Testing, refere-se a uma metodologia de teste de segurança cibernética projetada para avaliar a segurança de aplicações web e APIs em tempo de execução. Ao contrário das abordagens estáticas que analisam o código-fonte, o DAST simula ataques reais, interagindo dinamicamente com a aplicação como faria um usuário legítimo. Isso permite identificar vulnerabilidades que podem ser exploradas por atacantes externos, incluindo falhas como XSS (Cross-Site Scripting), SQL injection, CSRF (Cross-Site Request Forgery), entre outras.

1.9. Importância do DAST na Segurança Cibernética:

A metodologia DAST desempenha um papel fundamental ao oferecer uma visão prática e realista das vulnerabilidades de segurança presentes em aplicações web e APIs. Ao simular ataques em tempo real, as ferramentas de DAST ajudam a descobrir e mitigar vulnerabilidades antes que sejam exploradas por ameaças maliciosas, protegendo assim dados sensíveis, a reputação da empresa e a confiança dos usuários.

1.10. Objetivos da Metodologia DAST:

Os principais objetivos do DAST incluem não apenas a identificação de vulnerabilidades, mas também a integração contínua com fluxos de trabalho de desenvolvimento e operações (DevOps), a geração de relatórios detalhados para uma análise eficaz de riscos e a facilitação da conformidade com regulamentações de segurança cibernética.

Ao adotar a metodologia DAST como parte de uma estratégia abrangente de segurança cibernética, as organizações estão melhor equipadas para enfrentar os desafios emergentes no ambiente digital atual. Esta abordagem não apenas

fortalece a resistência contra ameaças cibernéticas, mas também promove uma cultura de segurança proativa e contínua, essencial para a proteção de dados e a sustentação da confiança dos clientes.

Ao longo deste documento, exploraremos detalhadamente como o DAST funciona, suas vantagens, desafios e melhores práticas para implementação eficaz, fornecendo assim um guia abrangente para integrar esta metodologia em estratégias de segurança de aplicações web e APIs.

1.11. Funcionamento Detalhado do DAST (Dynamic Application Security Testing)

O DAST é uma ferramenta de segurança cibernética projetada para identificar e mitigar vulnerabilidades em aplicações web e APIs através de testes dinâmicos. Ao contrário do SAST (Static Application Security Testing), que analisa o código-fonte sem executar o aplicativo, o DAST avalia a segurança da aplicação em tempo de execução, interagindo diretamente com ela como faria um usuário real. Abaixo estão os principais aspectos do funcionamento do DAST:

1.11.1. Configuração e Configurações Iniciais:

- **Configuração do Alvo**: O usuário configura o DAST para apontar para o URL da aplicação ou API que será testada.
- **Configurações de Autenticação**: Informações de autenticação podem ser fornecidas para que o DAST possa acessar áreas protegidas da aplicação durante o teste.

1.11.2. Exploração e Varredura Automatizada:

- **Varredura de Vulnerabilidades**: O DAST varre automaticamente todas as funcionalidades acessíveis da aplicação, explorando pontos de entrada como formulários web, links, parâmetros de URL e APIs expostas.

- **Injeção de Payloads**: Utiliza técnicas de injeção de payloads para testar vulnerabilidades, como XSS (Cross-Site Scripting), SQL Injection, Command Injection, entre outras.

1.11.3. Simulação de Ataques Realistas:

- **Comportamento do Usuário**: Interage com a aplicação como faria um usuário legítimo, enviando requisições HTTP e HTTPS, preenchendo formulários, clicando em links e submetendo dados.
- **Identificação de Respostas Anômalas**: Analisa as respostas recebidas da aplicação para identificar comportamentos anômalos que podem indicar uma vulnerabilidade de segurança.

1.11.4. Análise de Segurança em Tempo Real:

- **Detecção de Vulnerabilidades**: Durante a execução dos testes, o DAST detecta e classifica vulnerabilidades com base em padrões conhecidos e comportamentos inesperados.
- **Relatórios Detalhados**: Gera relatórios detalhados que descrevem cada vulnerabilidade encontrada, seu impacto potencial, a prova de conceito da exploração e recomendações para correção.

1.11.5. Análise de Segurança das Configurações do Servidor:

- **Configurações de Segurança**: Além de testar a aplicação em si, o DAST pode verificar a configuração do servidor web para identificar problemas como headers de segurança mal configurados, listagem de diretórios habilitada, ou permissões de arquivo inadequadas.

1.11.6. Integração com Ciclos de Desenvolvimento e Operações (DevOps):

- **Automatização e Integração Contínua**: Pode ser integrado a pipelines de CI/CD para realizar testes automáticos de segurança em cada iteração de código, garantindo que novas vulnerabilidades sejam identificadas antes da implantação em produção.

Benefícios do DAST (Dynamic Application Security Testing)

1.12. Identificação de Vulnerabilidades em Tempo Real:

- **Testes em Ambientes de Produção**: O DAST realiza varreduras durante a execução da aplicação, identificando vulnerabilidades que podem ser exploradas por atacantes reais.
- **Detecção de Falhas Críticas**: Identifica vulnerabilidades como XSS (Cross-Site Scripting), SQL Injection, falhas de autenticação, entre outras, que podem comprometer a segurança da aplicação.

1.13. Simulação de Ataques Realistas:

- **Comportamento de Atacante**: O DAST simula o comportamento de um atacante real, enviando requisições HTTP/HTTPS, explorando links, formulários e outras funcionalidades da aplicação para identificar pontos fracos.
- **Testes Abrangentes**: Avalia diferentes cenários de ataques que podem comprometer a integridade, confidencialidade e disponibilidade da aplicação.

1.14. Integração com Ciclos de Desenvolvimento (DevOps)

- **Automação e Integração Contínua**: Pode ser integrado a pipelines de CI/CD para executar testes de segurança de forma automatizada em cada iteração de código.
- **Feedback Imediato**: Fornecendo feedback rápido aos desenvolvedores sobre vulnerabilidades encontradas, facilitando correções rápidas e reduzindo o tempo de exposição a riscos de segurança.

1.15. Avaliação da Segurança do Ambiente de Produção

- **Configurações do Servidor**: Além de testar a aplicação, o DAST verifica configurações de segurança do servidor web, como headers HTTP, configurações de TLS/SSL e permissões de arquivo, identificando configurações inadequadas que podem expor a aplicação a riscos.

1.16. Relatórios Detalhados e Recomendações de Correção

- **Documentação Clara de Vulnerabilidades**: Gera relatórios detalhados que descrevem cada vulnerabilidade encontrada, seu impacto potencial e recomendações específicas para correção.
- **Priorização de Correções**: Ajuda na priorização de esforços de correção com base na severidade das vulnerabilidades identificadas, permitindo que equipes de segurança e desenvolvimento foquem nos problemas mais críticos primeiro.

1.17. Suporte a Conformidade e Regulamentações

- **Atendimento a Requisitos Regulatórios**: Ajuda as organizações a atenderem a padrões de segurança e regulamentações como PCI-DSS, GDPR, entre outros, garantindo que as aplicações estejam em conformidade com os requisitos de segurança exigidos.

1.18. Testes Contínuos de Segurança:

- **Monitoramento e Manutenção**: Oferece suporte ao monitoramento contínuo de segurança, permitindo que as organizações mantenham a proteção das aplicações mesmo após o lançamento inicial.

Exemplos de Análises Realizadas pelo DAST

1.19. Identificação de Vulnerabilidades de Injeção de SQL

- **Descrição**: O DAST simula ataques de injeção de SQL, enviando inputs maliciosos através de formulários web, parâmetros de URL ou campos de entrada. Ele verifica se a aplicação manipula corretamente esses inputs e se protege contra consultas SQL não autorizadas.
- **Exemplo**: Identifica vulnerabilidades onde um atacante pode manipular queries SQL para acessar, modificar ou excluir dados no banco de dados.

1.20. Teste de XSS (Cross-Site Scripting)

- **Descrição**: O DAST envia payloads de XSS através de entradas de usuário, como campos de formulários, cookies ou headers HTTP. Ele verifica se a aplicação filtra corretamente inputs que podem conter scripts maliciosos.

- **Exemplo**: Identifica vulnerabilidades onde scripts maliciosos podem ser injetados e executados no navegador de usuários, potencialmente comprometendo a segurança da aplicação.

1.21. Análise de Vulnerabilidades de Autenticação e Autorização

- **Descrição**: O DAST testa as funcionalidades de autenticação e autorização da aplicação, enviando requisições com diferentes credenciais e níveis de acesso. Ele verifica se a aplicação impede o acesso não autorizado a recursos protegidos.

- **Exemplo**: Identifica falhas como autenticação fraca, cookies de sessão não seguros, ou falhas na implementação de políticas de controle de acesso.

1.22. Exploração de Vulnerabilidades de CSRF (Cross-Site Request Forgery)

- **Descrição**: O DAST simula ataques de CSRF enviando solicitações maliciosas a partir de um site confiável onde o usuário autenticado está logado. Ele verifica se a aplicação valida corretamente a origem das solicitações.

- **Exemplo**: Identifica vulnerabilidades onde um atacante pode enganar um usuário autenticado para executar ações não intencionais na aplicação.

1.23. Varredura de Segurança em APIs

- **Descrição**: O DAST testa a segurança de APIs, enviando requisições com payloads maliciosos para endpoints expostos. Ele verifica se as APIs estão adequadamente protegidas contra ataques como injeção de parâmetros ou acesso não autorizado.

- **Exemplo**: Identifica vulnerabilidades onde dados sensíveis são expostos através de APIs mal configuradas ou endpoints não protegidos adequadamente.

1.24. Análise de Segurança das Configurações do Servidor Web

- **Descrição**: Além de testar a aplicação em si, o DAST verifica configurações de segurança do servidor web, como headers HTTP, configurações TLS/SSL e permissões de arquivo.
- **Exemplo**: Identifica configurações inadequadas que podem expor a aplicação a riscos de segurança, como listagem de diretórios, ou headers de segurança mal configurados.

2. Exemplos de Ferramentas SAST

Burp Suite Professional:

- **Descrição**: Burp Suite é uma ferramenta de teste de segurança amplamente usada que oferece módulos específicos para DAST. Ele permite varreduras automatizadas e manuais de aplicações web para identificar vulnerabilidades como XSS, injeção SQL, e outras.
- **Características**: Varredura automatizada, exploração de vulnerabilidades, relatórios detalhados, e suporte à integração com CI/CD.

Netsparker:

- **Descrição**: Netsparker é uma ferramenta automatizada de segurança de aplicações web que realiza varreduras DAST para identificar e relatar vulnerabilidades de segurança.

- **Características**: Varredura automática, identificação de vulnerabilidades como XSS, injeção SQL, e outros, integração com CI/CD, e relatórios detalhados.

AppScan:

- **Descrição**: IBM Security AppScan é uma ferramenta robusta para testes de segurança de aplicações web e serviços web. Ela oferece varreduras DAST para identificar e corrigir vulnerabilidades de segurança.

- **Características**: Varredura automatizada, suporte a APIs RESTful e SOAP, relatórios de conformidade, e integração com ferramentas de desenvolvimento.

Acunetix:

- **Descrição**: Acunetix é uma ferramenta de varredura de segurança de aplicações web que inclui funcionalidades de DAST para identificar vulnerabilidades como XSS, SQL injection, e outros.

- **Características**: Varredura automática, exploração de vulnerabilidades, relatórios detalhados, e suporte a integração com CI/CD.

Qualys Web Application Scanning (WAS):

- **Descrição**: Qualys WAS é uma plataforma de segurança de aplicações baseada em nuvem que realiza varreduras DAST para identificar e relatar vulnerabilidades em aplicações web.

- **Características**: Varredura automática, identificação de vulnerabilidades como XSS, injeção SQL, e outros, relatórios detalhados, e integração com plataformas de gerenciamento de vulnerabilidades.

Essas ferramentas são projetadas para ajudar organizações a identificar e corrigir vulnerabilidades de segurança em aplicações web e APIs de forma eficiente e eficaz. Elas oferecem recursos robustos para automação de testes, análise de resultados detalhados e integração com fluxos de trabalho de desenvolvimento e operações, proporcionando uma abordagem proativa para proteção contra ameaças cibernéticas.

Benefícios do DAST

- **Detecção de Vulnerabilidades em Tempo Real:**

Testa a aplicação em seu ambiente de produção, identificando vulnerabilidades que podem ser exploradas por atacantes reais.

- **Simulação de Ataques Realistas:**

Comporta-se como um invasor legítimo, enviando requisições HTTP/HTTPS e explorando diferentes pontos de entrada da aplicação.

- **Integração com Ciclos de Desenvolvimento (DevOps):**

Facilita a integração contínua e a automação de testes de segurança, fornecendo feedback imediato aos desenvolvedores.

- **Análise de Segurança das Configurações do Servidor:**

Verifica configurações do servidor web que podem afetar a segurança da aplicação.

- **Relatórios Detalhados e Recomendações de Correção:**

Oferece insights claros sobre cada vulnerabilidade identificada, ajudando na priorização e correção rápida.

Ao adotar o DAST como parte de uma estratégia abrangente de segurança cibernética, as organizações podem mitigar riscos significativos e fortalecer suas defesas contra ameaças digitais. A combinação do DAST com outras práticas, como análise estática de código (SAST) e monitoramento contínuo de segurança, é essencial para garantir que aplicações permaneçam seguras em um ambiente digital dinâmico e desafiador.

Em suma, investir em ferramentas como o DAST não apenas protege as aplicações contra vulnerabilidades conhecidas, mas também ajuda a evitar potenciais violações de dados e danos à reputação da organização. Ao priorizar a segurança desde as fases iniciais de desenvolvimento até a manutenção contínua, as empresas podem garantir a confiança dos usuários e clientes, além de estar em conformidade com regulamentações de segurança cibernética cada vez mais rigorosas.

Importância das Ferramentas de DAST

- **Identificação Proativa de Vulnerabilidades**: Permitem às organizações identificar vulnerabilidades como XSS, SQL injection, CSRF, entre outras, antes que sejam exploradas por atacantes mal-intencionados.
- **Realismo na Simulação de Ataques**: Ao comportarem-se como um invasor legítimo, as ferramentas de DAST proporcionam uma visão precisa de como a aplicação pode ser explorada em um ambiente de produção.
- **Integração com Fluxos de Trabalho de Desenvolvimento**: Facilitam a integração contínua (CI) e a entrega contínua (CD), fornecendo feedback rápido aos desenvolvedores sobre vulnerabilidades descobertas durante o desenvolvimento e testes.

- **Relatórios Detalhados e Recomendações de Correção**: Oferecem relatórios detalhados que descrevem cada vulnerabilidade encontrada, seu potencial de impacto e orientações claras sobre como corrigi-las.

- **Suporte à Conformidade e Regulamentações**: Ajudam as organizações a cumprir requisitos regulatórios e padrões de segurança, garantindo que suas aplicações estejam em conformidade com normas como PCI-DSS, GDPR, entre outras.

Conclusão sobre Ferramentas de DAST

Investir em ferramentas de DAST não é apenas uma medida preventiva, mas uma prática essencial para proteger ativos digitais e dados sensíveis contra ameaças em um cenário cibernético cada vez mais sofisticado. Ao implementar e integrar essas ferramentas como parte de uma estratégia de segurança abrangente, as organizações não só melhoram a segurança de suas aplicações, mas também fortalecem a confiança dos usuários e clientes.

Portanto, ao escolher uma ferramenta de DAST, é crucial considerar suas funcionalidades, capacidades de integração, suporte e relatórios oferecidos para garantir uma proteção eficaz contra ameaças digitais. Com a adoção adequada dessas ferramentas, as organizações estão melhor posicionadas para enfrentar os desafios de segurança cibernética e mitigar riscos de maneira proativa.

Sobre o Autor

Kleber Melo – CISSP, DPO, ISO 27001 Lead Auditor

CEO da Mindsec

Redator Chefe do Blog Minuto da Segurança

Sócio Fundador da Mindsec, empresa especializada em Segurança e Privacidade com mais de 30 anos de experiência nas áreas de Tecnologia, Segurança da Informação, Continuidade de Negócios, Cyber Security e Análise e Gestão de Risco de Segurança.

Atuante no mercado de Segurança da Informação em empresas como IBM, Banco Sudameris/ABN, Banco HSBC e Banco Original. No HSBC atuou como Gerente Regional de Segurança da Informação América Latina do Banco HSBC, responsável pela condução da estratégica de SI para 15 países e líder global para direcionamento, seleção, escolha e implementação de software/processos de segurança global.

Professor de cursos de pós-graduação de diversas disciplinas de segurança nas universidades Mackenzie, IPEN-USP, FGV e IBTA

Graduação em Matemática Aplicada à Informática e Mestrado em engenharia da Computação pela Universidade Mackenzie

Mindsec

A Mindsec, com mais de 12 ANOS de experiência, procura trazer ao mercado uma forma integrada e adaptativa de fazer Segurança da Informação.

A proteção da informação, não pode ser uma aplicação pura e simples de softwares e normas, mas uma forma inteligente e integrada com os objetivos de negócio. Desta forma, auxiliamos na estruturação de ações flexíveis de segurança da informação, de forma a obter a proteção mais adequada para o seu negócio, como o *melhor ROI*, o *melhor desempenho* e a *maior agilidade*.

Seguindo os conceitos de uma Information ***Security Service & Solution***, propiciamos às organizações o melhor custo-benefício para o seu planejamento tecnológico, quanto a proteção da informação, privacidade de dados e plano de continuidade de negócios, nos seus projetos de segurança, tecnologia, LGPD e transformação digital.