Detectando Vulnerabilidades em Aplicações Implementando testes SAST e DAST na prática!

Renato Groffe
Microsoft MVP, MTAC
linkedin.com/in/renatogroffe
renatogroffe.medium.com

Microsoft® Most Valuable Professional

MTAC org.br

APISEC UNIVERSITY AMBASSADOR

DEVPIRA

Renato Groffe

- Microsoft Most Valuable Professional (MVP)
- Docker Captain
- APISEC U Ambassador
- Multi-Plataform Technical Audience Contributor (MTAC)
- Arquiteto de Soluções/Software
- +20 anos de experiência na área de Tecnologia
- Community Leader, Autor Técnico e Palestrante

Certificações Gratuitas em Segurança

www.apisecuniversity.com/#courses

APISEC
UNIVERSITY

Conteúdos desta apresentação

Deixem um
star ⭐ apoiando

github.com/renatogroffe/sast-dast_devpira-festival-2025

Agenda

- SAST e DAST: uma visão geral
- Ferramentas para automação de testes de segurança
- Exemplos práticos

Por que analisar a segurança do código?

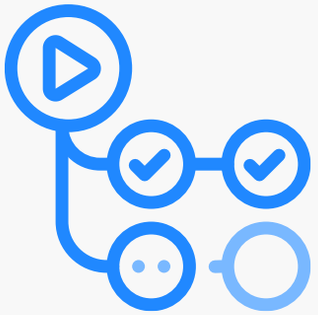
- Identificar de forma precoce vulnerabilidades
- Evitar a exposição de informações sensíveis via código
- Diminuir o risco de ataques
- Reduzir custos

Alguns problemas comuns...

- Dependências desatualizadas e que podem levar a vulnerabilidades
- Configurações e secrets expostos
- APIs com problemas de autorização/autenticação

Como implementar segurança do código?

- Implementar práticas baseadas em **SAST (Static Application Security Testing)**
- Uso de automação sempre que possível (**Azure DevOps**, **GitHub Actions**, **GitLab**...)
- Existe um **formato padronizado** para a apresentação de resultados → **SARIF (Static Analysis Results Interchange Format)**



E quanto a uma aplicação em execução?

- Aplicações Web podem expor endpoints que seriam explorados em ataques

- Problemas como **Cross-Origin Resource Sharing (CORS)**, **Server-Side Request Forgery (SSRF)** e **exposição excessiva de dados** também são portas de entrada de ataques

- Adoção de práticas baseadas em **DAST (Dynamic Application Security Testing)**

DAST x Pen Tests

- Ambos testarão **aplicações em execução**
- Testes do tipo **DAST** requerem sempre o uso de **ferramentas de automação**, possibilitando com isto sua integração com esteiras de **CI/CD**
- **Penetration Tests** envolvem tanto o uso de automação, quanto checagens manuais

Algumas soluções que envolvem licenciamento

sonarqube

snyk

GitHub Advanced Security

Checkmarx

VERACODE

...

Algumas soluções gratuitas/open source

DEPENDENCY-CHECK

checkov

Kubescape

aqua trivy

KICS by Checkmarx

APIsec Scanner: implementando DAST

- Alternativa para **DAST**
- Testes de **sites** e **APIs REST** em execução
- **Opção gratuita para testes mais simplificados**
- Site: https://www.apisec.ai/

OWASP Dependency-Check: packages e bibliotecas

- Análise de vulnerabilidades em **pacotes** e **bibliotecas** utilizadas por projetos de software
- Requer o uso de uma chave do **National Vulnerability Database**, projeto mantido pelo **NIST - National Institute of Standards and Technology** (órgão do governo norte-americano)
- Suporte a **múltiplas plataformas de desenvolvimento**
- Site: https://owasp.org/www-project-dependency-check/

Trivy: uma alternativa para containers

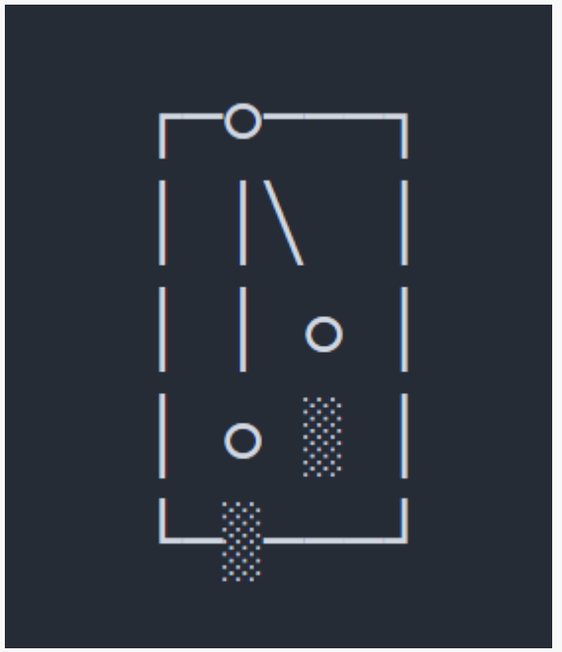
- Scanning de **imagens**, **Dockerfiles** e **arquivos**
- Identificação de vulnerabilidades como **dependências desatualizadas** (**aplicação** e **sistema operacional**), **configurações** das imagens e até mesmo alguns tipos de **secrets**
- Projeto **open source** mantido pela **Acqua Security**
- Site: https://trivy.dev/

Docker Scout

- Análise de vulnerabilidades em imagens de containers
- Similar ao projeto **Trivy**
- Execução via **CLI** ou **container**
- Resultados podem ser publicados em formatos como **SARIF (Static Analysis Results Interchange Format)** e **Markdown**
- Site: https://docs.docker.com/scout/

gitleaks: encontrando segredos no código

- Também **open source**
- Capacidade de **varrer toda uma estrutura de diretórios** e apontar secrets/credenciais encontrados
- Site: https://github.com/gitleaks/gitleaks



Checkov: testando seu código de infraestrutura

- Scanning voltado a **IaC (Infrastructure as Code)**
- **Suporte a Kubernetes (YAML), Terraform, AWS Cloud Formation, Azure ARM Templates, Dockerfile…**
- Projeto **open source** mantido pela **Prisma Cloud**
- Site: https://www.checkov.io/

checkov

KICS: uma alternativa para IaC

- Scanning voltado a **IaC (Infrastructure as Code)**
- Suporte a Kubernetes (YAML), Terraform, AWS Cloud Formation, Azure ARM Templates, Dockerfile, OpenAPI...
- Projeto **open source** mantido pela **Checkmarx**
- Site: https://www.kics.io/

Vulnerabilidades em clusters Kubernetes

- Análises detectando falhas de configuração e vulnerabilidade
- Popeye: https://github.com/derailed/popeye
- Kubescape: https://kubescape.io/

Vulnerabilidades em Apps Mobile: MobSF

- Mobile Security Framework (MobSF)
- Análises detectando problemas de implementação, sistemas operacionais (**Android** e **iOS**), packages desatualizados…
- Site: https://github.com/MobSF/Mobile-Security-Framework-MobSF

EXEMPLOS PRÁTICOS

Conteúdos desta apresentação

Deixem um
star ⭐ apoiando

github.com/renatogroffe/sast-dast_devpira-festival-2025

Certificações Gratuitas em Segurança

www.apisecuniversity.com/#courses

OBRIGADO!