Análise Comparativa: Ferramentas SAST versus Modelos de IA para Detecção de Vulnerabilidades

Data: 10 de outubro de 2025 **Projetos Avaliados:** DVPWA e PyGoat **Ferramentas SAST:** Bandit, Semgrep, Semgrep Pro **Modelos de IA:** Gemini 2.5 Pro, Claude Sonnet 4.5, GPT-5, Grok-4

Resumo Executivo

Esta pesquisa avaliou comparativamente ferramentas de análise estática de segurança (SAST) e modelos de linguagem de IA na detecção de vulnerabilidades em aplicações Python. Utilizando groundtruths de 48 vulnerabilidades (DVPWA) e 24 vulnerabilidades (PyGoat), calculamos métricas de Verdadeiros Positivos (TP), Falsos Positivos (FP) e Falsos Negativos (FN) para determinar precisão, recall e F1 score.

Principais Descobertas:

- Ferramentas SAST apresentam precisão perfeita (100%) mas recall crítico (2-4%), detectando apenas uma pequena fração das vulnerabilidades
- Modelos de IA demonstram recall 4-7x superior, porém com precisão variável (27-100%)
- Desempenho varia drasticamente entre diferentes codebases, indicando dependência contextual
- Nenhuma ferramenta fornece cobertura adequada isoladamente

Metodologia

Abordagem de Avaliação

Comparamos cada ferramenta contra groundtruths curados, utilizando correspondência de localização (arquivo:linha) e validação CWE quando disponível. As métricas calculadas foram:

- **Precisão** = TP / (TP + FP) — Acurácia dos achados reportados
- **Recall** = TP / (TP + FN) — Completude da detecção (cobertura do groundtruth)
- **F1 Score** = 2 × (Precisão × Recall) / (Precisão + Recall) — Métrica balanceada

Nota sobre Verdadeiros Negativos (TN): Excluímos TN da análise, pois em detecção de vulnerabilidades isso requereria enumerar todas as possíveis localizações não-vulneráveis — um espaço indefinido ou infinito.

Projetos Analisados

- **DVPWA (Damn Vulnerable Python Web Application):** 48 vulnerabilidades documentadas
- **PyGoat:** 24 vulnerabilidades documentadas

Resultados

DVPWA: Melhor Cenário para Ferramentas Automatizadas

Ferramenta	TP	FP	FN	Precisão	Recall	F1
SAST Tools						
bandit	2	0	46	100%	4,17%	0,08
semgrep	2	0	46	100%	4,17%	0,08
semgrep-pro	1	0	47	100%	2,08%	0,04
AI Models						
gemini-2.5-pro	4	2	44	66,67%	8,33%	0,15
claude-4.5-sonnet	8	3	40	72,73%	16,67%	0,27
gpt-5	13	1	35	92,86%	27,08%	0,42
grok-4	**14**	**0**	**34**	**100%**	**29,17%**	**0,45**

Análise DVPWA:

- Ferramentas SAST mantiveram precisão perfeita mas detectaram <5% das vulnerabilidades
- Grok-4 alcançou o melhor desempenho geral: precisão perfeita com recall 7x superior ao SAST
- GPT-5 demonstrou equilíbrio notável (92,86% precisão, 27% recall)
- Mesmo o melhor modelo deixou 71% das vulnerabilidades sem detecção

PyGoat: Cenário Desafiador com Colapso de Performance

Ferramenta	TP	FP	FN	Precisão	Recall	F1
bandit	1	58	23	1,69%	4,17%	0,02
semgrep	0	1	24	0%	0%	0
semgrep-pro	0	1	24	0%	0%	0
gemini-2.5-pro	**8**	**22**	**16**	**26,67%**	**33,33%**	**0,30**
claude-sonnet-4-5	1	24	23	4%	4,17%	0,04
gpt-5	2	38	22	5%	8,33%	0,06
grok-4-0709	1	9	23	10%	4,17%	0,06

Análise PyGoat:

- **Colapso generalizado de precisão:** Maioria das ferramentas <10% (vs 67-100% em DVPWA)
- **Bandit gerou 58 falsos positivos** em 59 achados — taxa de ruído inaceitável
- **Gemini emergiu como melhor performer** neste contexto (26,67% precisão, 33,33% recall)
- Desempenho dramaticamente diferente entre projetos evidencia dependência contextual

---

Discussão

O Dilema Precisão-Recall

Ferramentas SAST adotam uma estratégia conservadora: reportam apenas achados de altíssima confiança, garantindo que desenvolvedores confiem em cada alerta. Isso resulta em **precisão perfeita mas recall crítico** — detectam corretamente o que encontram, mas encontram muito pouco.

Modelos de IA demonstram o trade-off oposto: capacidade semântica superior permite identificar mais vulnerabilidades, mas com risco de falsos positivos. No DVPWA, modelos como Grok-4 e GPT-5 mantiveram precisão >90% enquanto detectaram 7x mais vulnerabilidades que ferramentas SAST.

Experiência do Desenvolvedor e Fadiga de Alertas

Limiares de Confiança:

- **>95% precisão:** Alta confiança, alertas levados a sério
- **90-95% precisão:** Aceitável com boa UX
- **80-90% precisão:** Frustração começa, especialmente com alto volume
- **<80% precisão:** Credibilidade da ferramenta colapsa

No PyGoat, a maioria das ferramentas ficou abaixo de 10% de precisão, gerando entre 9-58 falsos positivos. Em bases de código maiores, isso representaria centenas de alertas incorretos, causando fadiga severa e levando desenvolvedores a ignorarem até mesmo achados legítimos.

O Falso Senso de Segurança

Baixo recall é mais perigoso que baixa precisão. Enquanto falsos positivos geram frustração operacional, falsos negativos criam vulnerabilidade real:

1. **Vulnerabilidades não-detectadas vão para produção:** 34-47 vulnerabilidades críticas no DVPWA não foram encontradas
2. **Teatro de compliance:** Organizações marcam "executamos ferramentas de segurança" sem alcançar segurança real
3. **Recursos mal alocados:** Equipes focam em corrigir 2-14 problemas detectados enquanto 34-46 outros permanecem
4. **Falsa confiança:** Stakeholders acreditam que a aplicação é segura porque "as ferramentas não encontraram problemas"

Variabilidade Entre Codebases

A diferença dramática de desempenho entre DVPWA e PyGoat revela uma limitação fundamental: **ferramentas automatizadas são altamente sensíveis ao contexto**. Fatores que influenciam:

- Tipos de vulnerabilidades presentes (injeção SQL vs falhas lógicas)
- Padrões de código e arquitetura
- Framework e biblioteca utilizados
- Definição e granularidade do groundtruth
- Complexidade e sofisticação das vulnerabilidades

Isso invalida a noção de uma "ferramenta universal" — diferentes projetos requerem diferentes abordagens.

Conclusões e Recomendações

Conclusões Principais

1. **Nenhuma ferramenta fornece cobertura adequada isoladamente** — mesmo o melhor performer detectou <30% das vulnerabilidades
2. **Ferramentas SAST são confiáveis mas insuficientes** — precisão perfeita com recall de 2-4%
3. **Modelos de IA mostram potencial mas precisam refinamento** — recall 4-7x superior mas com desafios de precisão
4. **Desempenho varia drasticamente entre contextos** — resultados não são generalizáveis
5. **Baixo recall cria falso senso de segurança** — mais perigoso que falsos positivos

Recomendações Estratégicas

Para Equipes de Segurança:

- Implementar abordagem em camadas: SAST + IA + revisão manual + testes de penetração
- Medir e reportar recall, não apenas "achados remediados"
- Estabelecer expectativas realistas sobre capacidades de ferramentas automatizadas
- Investir em treinamento de desenvolvedores sobre segurança

Para Equipes de Desenvolvimento:

- Não ignorar achados de IA reflexivamente — precisão >90% significa que maioria é real
- Fornecer feedback sobre falsos positivos para refinamento de modelos
- Entender lacunas — ferramentas aprovadas ≠ código seguro
- Participar ativamente de revisões de segurança

Para Seleção de Ferramentas:

- **Ferramentas SAST:** Essenciais para gates de CI/CD; priorizar por cobertura de linguagem/framework
- **Modelos de IA:** Valiosos para revisão de código; escolher baseado em precisão e qualidade de integração
- **Abordagem híbrida:** Usar SAST para bloqueio automatizado, IA para achados consultivos
- **Avaliação contínua:** Reavaliar regularmente efetividade contra seu landscape real de vulnerabilidades

Limitações do Estudo

- Groundtruths podem estar incompletos ou conter erros
- Avaliação em apenas duas aplicações deliberadamente vulneráveis
- Resultados podem não se generalizar para código de produção
- Desempenho de modelos de IA depende fortemente de configuração e prompts
- Ferramentas evoluem rapidamente — resultados são snapshot temporal

---

Implicações para a Indústria

Esta pesquisa evidencia que estamos nos estágios iniciais da detecção automatizada de vulnerabilidades. A lacuna entre "o que ferramentas podem encontrar" e "o que vulnerabilidades realmente existem"

permanece enorme — mesmo ferramentas de ponta detectam menos de um terço dos problemas de segurança.

Para organizações sérias sobre segurança, a resposta não é escolher entre SAST e IA — é integrar ambos criteriosamente, reconhecendo que ferramentas automatizadas são **um componente** de um programa de segurança abrangente que deve incluir revisão manual, análise arquitetural, modelagem de ameaças e treinamento contínuo de segurança.

O equilíbrio entre produtividade do desenvolvedor e eficácia de segurança — gerenciando fadiga de alertas enquanto maximiza detecção de vulnerabilidades — permanece o desafio central da engenharia de ferramentas de segurança de aplicações.