ANÁLISE DA SEGURANÇA DE APLICAÇÕES MÓVEIS POR MEIO DO USO DE TESTES DE INVASÃO[1]

Fernando Brito Campideli (PIC), Luciana Andréia Fondazzi Martimiano (Orientadora).
E-mail: lafmartimiano@uem.br

Universidade Estadual de Maringá, Centro de Tecnologia, Maringá, PR.

Área e subárea do conhecimento: Engenharias/Ciência da Computação.

Palavras-chave: OWASP; Pentest; Análise de Vulnerabilidades.

RESUMO

O avanço no ecossistema de dispositivos móveis consolidou os aplicativos como canais essenciais para o tráfego de dados sensíveis e serviços críticos. Este trabalho apresenta uma avaliação prática de segurança em ambiente mobile, fundamentada nas diretrizes internacionais da *Open Worldwide Application Security Project* (OWASP), incluindo o OWASP Mobile Top 10 e o *Mobile Application Security Testing Guide* (MASTG). Por meio de técnicas de análises estática e dinâmica, o aplicativo móvel foi avaliado. Os testes práticos evidenciaram superfícies críticas de ataque decorrentes da arquitetura híbrida (*wrapper* WebView), como exposição de chaves de API, vazamento de Informações de Identificação Pessoal (PII) em parâmetros de URL (*Uniform Resource Locator*), fragilidades no ciclo de autenticação JWT (*JSON Web Token*) e rotas suscetíveis à enumeração de usuários. A aplicação estruturada e combinada de metodologias padronizadas de testes de intrusão permite validar a integridade dos controles de segurança, mapeando vetores de risco e fornecendo subsídios técnicos indispensáveis para a mitigação de vulnerabilidades e o fortalecimento das defesas institucionais.

INTRODUÇÃO

O uso massivo de dispositivos móveis consolidou esses aparelhos como o principal meio de acesso à internet, concentrando grande volume de dados sensíveis e transações institucionais (FORTINET, 2026). No entanto, o ecossistema móvel frequentemente opera sob ciclos de desenvolvimento dinâmicos que exigem validação constante de suas salvaguardas de segurança (CONVISO APPSEC, 2015). A ausência de conformidade com padrões consolidados expõe as aplicações a riscos severos, como roubo de credenciais, quebra de privacidade e danos à integridade do sistema operacional e de servidores de *backend* (BLAZE INFORMATION SECURITY, 2025).

[1]Por motivos de segurança cibernética e divulgação responsável de vulnerabilidades, a identificação nominal do aplicativo analisado foi omitida no resumo e no corpo do texto.

Nesse cenário, os testes de invasão (*pentest*) estabelecem-se como mecanismos indispensáveis para simular ataques reais, permitindo avaliar a eficácia dos controles defensivos e identificar falhas estruturais antes de sua exploração por agentes maliciosos (IBM, 2024). Para sistematizar essas análises, a comunidade técnica adota as diretrizes da *Open Worldwide Application Security Project* (OWASP), destacando-se o OWASP Mobile Top 10, o *Mobile Application Security Verification Standard* (MASVS) e o *Mobile Application Security Testing Guide* (MASTG) (OWASP, 2024). Este estudo analisa e aplica os procedimentos técnicos de testes de intrusão voltados a aplicações móveis, validando a resiliência e a segurança da infraestrutura.

MATERIAIS E MÉTODOS

A condução do estudo estruturou-se sob a metodologia sistemática proposta pelo OWASP MASTG e pelas etapas clássicas de teste de intrusão: levantamento de informações e reconhecimento; modelagem de ameaças e análise da superfície de ataque; exploração ativa e validação de controles de acesso; e consolidação dos achados para mitigação (IBM, 2024).

O ambiente experimental foi configurado sobre distribuição Linux com emulação Android via Genymotion com privilégios de superusuário (*root*), integrando o seguinte arcabouço técnico e procedimental:

- **Análise Estática (SAST) e Engenharia Reversa:** Extração do pacote APK do ambiente emulado via Android Debug Bridge (`adb pull`) e descompilação utilizando *Apktool*, *Jadx* e *Mobile Security Framework* (MobSF) em contêiner Docker. O procedimento abrangeu a inspeção do manifesto (`AndroidManifest.xml`), mapeamento de permissões, busca por chaves de API e segredos embutidos, além da varredura de *strings* nos binários nativos compilados em C/C++ (`libapp.so`) com utilitários `strings` e `grep`;
- **Configuração de Rede e Bypass de SSL Pinning:** Estabelecimento de servidor proxy interceptador no Burp Suite (porta 8083) com modo *invisible proxying*. Para superar falhas de *handshake* TLS (*SSLHandshakeException*) e restrições de certificados autoassinados pela biblioteca nativa Chromium do WebView, o certificado da Autoridade Certificadora (CA) do Burp Suite foi convertido no formato de *hash* de assunto (*subject hash*) e injetado diretamente no diretório do sistema restrito do Android (`/system/etc/security/cacerts/`);
- **Análise Dinâmica (DAST) e Fuzzing de API:** Monitoramento e manipulação de tráfego HTTP/HTTPS em tempo de execução. Foram realizados testes de injeção de parâmetros (*fuzzing*), análise da estrutura de *tokens* de sessão, interceptação de chamadas assíncronas do *framework* web e tentativas de controle de acesso quebrado (*Broken Object Level Authorization* – BOLA/IDOR).

RESULTADOS E DISCUSSÃO

A aplicação integrada dos métodos revelou que a aplicação atua arquiteturalmente como um *wrapper* desenvolvido em Flutter que renderiza páginas e rotas institucionais gerenciadas pelo ZK Framework. Na etapa estática com o MobSF e inspeção de binários, constatou-se a exposição de uma Google API Key e a referência a um banco Firebase Realtime Database, o qual se encontrava com resposta HTTP 423 (*Locked*) após o encerramento do processo seletivo.

A análise dinâmica do tráfego e dos fluxos de autenticação evidenciou os seguintes achados técnicos:

- **Fragilidade na Emissão de JWT (CWE-287):** O *endpoint* respondeu com código HTTP 200 e mensagem de credenciais inválidas perante entradas incorretas, porém emitiu e retornou no corpo JSON um *token* JWT válido (algoritmo HS512). O *payload* revelou artefatos de homologação em ambiente de produção (*issuer* configurado como "teste"), ausência da *claim* de sujeito (sub) e tempo de expiração prolongado de 24 horas;
- **Exposição de Credenciais em Trânsito (CWE-319):** Durante a navegação no menu do usuário, constatou-se o envio do *hash* MD5 da senha do usuário em texto claro diretamente na *query string* (parâmetro chave) e em cabeçalhos Referer, expondo o segredo em logs intermediários de rede;
- **Vazamento de PII (CWE-359):** A URL retornada expôs o Cadastro de Pessoas Físicas (CPF) do usuário em texto claro na estrutura da requisição;
- **Enumeração de Usuários (CWE-204 e CWE-307):** Na rota de login e recuperação de senha, a ausência de mecanismos anti-automação (como CAPTCHA ou *rate limiting*) combinada a discrepâncias nas mensagens de erro do *backend* viabilizou a enumeração e confirmação de números de inscrição válidos na base de dados.

Por outro lado, os testes de manipulação de parâmetros horizontais (BOLA/IDOR) e chamadas assíncronas de componentes (POST /zkau) confirmaram a robustez de controles específicos no *backend*. A alteração de identificadores de candidatos em requisições GET foi bloqueada pelo servidor devido à validação cruzada entre o número de inscrição e o *hash* da senha. Adicionalmente, as chamadas assíncronas do ZK Framework mantiveram a segregação de contexto atrelada à sessão *stateful* (JSESSIONID e identificador de componente), impedindo a contaminação de dados entre sessões distintas. A convergência metodológica comprovou que a aplicação sistemática de testes estáticos e dinâmicos proporciona uma avaliação abrangente e precisa da postura de segurança da aplicação.

CONCLUSÕES

A execução estruturada de testes de invasão orientada pelos referenciais OWASP valida com precisão a resiliência e a integridade de aplicações móveis. A condução prática sobre o ecossistema do aplicativo analisado mostrou como uma abordagem metodológica multifacetada — abrangendo desde a descompilação de binários até a interceptação e manipulação de tráfego em tempo de execução — permite mapear falhas arquiteturais críticas (como vazamento de PII e fragilidades de autenticação) e confirmar a eficácia dos controles de estado do servidor. Conclui-se que o emprego rigoroso e padronizado dessas metodologias consolida a visibilidade sobre a superfície de ataque, viabilizando o endurecimento contínuo da infraestrutura e a proteção dos dados dos usuários.

REFERÊNCIAS

BLAZE INFORMATION SECURITY. **Pentest em aplicativos mobile: tudo o que precisa saber**. [2025]. Disponível em: https://www.blazeinfosec.com/pt-br/post/pentest-em-aplicativos-mobile-tudo-o-que-precisa-saber/. Acesso em: 13 maio 2026.

CERZOSIMO, R. F.; SILVA, C. A. da. **Vulnerabilidades em aplicativos mobile Android®**. Trabalho de Conclusão de Curso (Engenharia de Computação) – Faculdade de Computação, Universidade Federal de Mato Grosso do Sul, Campo Grande, 2024.

CONVISO APPSEC. **Tudo o que você precisa saber sobre ataques em aplicações mobile**. [2015]. Disponível em: https://blog.convisoappsec.com/tudo-o-que-voce-precisa-saber-sobre-ataques-em-aplicacoes-mobile/. Acesso em: 13 maio 2026.

FORTINET. **Segurança de aplicativos móveis**. [2026]. Disponível em: https://www.fortinet.com/br/resources/cyberglossary/mobile-app-security. Acesso em: 13 maio 2026.

IBM. **O que é teste de penetração?**. [2024]. Disponível em: https://www.ibm.com/br-pt/think/topics/penetration-testing. Acesso em: 13 maio 2026.

OWASP. **OWASP Mobile Top 10**. [2024]. Disponível em: https://owasp.org/www-project-mobile-top-10/. Acesso em: 13 maio 2026.