Vulnerabilidades em aplicativos mobile Android®

Raquel Freire Cerzosimo[1], **Carlos Alberto da Silva**[2]

[1]Curso de Engenharia de Computação – Faculdade de Computação (FACOM)
Universidade Federal de Mato Grosso do Sul (UFMS).
Av. Costa e Silva, s/n. - Bairro Universitário - CEP 79070-900 - Campo Grande - MS, Brasil.

r.freire@ufms.br, carlos.silva@ufms.br

Abstract. *This work explores security in mobile applications through penetration testing (pentest), highlighting the Open Web Application Security Project Mobile Application Security (OWASP MAS) methodology. The increasing reliance on mobile devices and applications makes protection against cyber threats crucial, especially considering the sensitivity of personal data stored and processed by these applications.*

Resumo. *Este trabalho explora a segurança em aplicativos móveis por meio de testes de penetração (pentest), destacando a metodologia* Open Web Application Security Project Mobile Application Security *(OWASP MAS). A crescente dependência de dispositivos móveis e aplicativos torna crucial a proteção contra ameaças cibernéticas, especialmente considerando a sensibilidade dos dados pessoais armazenados e processados por essas aplicações.*

1. Introdução

Quando fala-se de segurança em tecnologia existem inúmeros universos a serem explorados. A primeira escolha que deve ser feita é em qual tipo de dispositivo: notebooks, computadores, celulares, embarcados, etc. Independente da plataforma escolhida, o processo de se verificar que o dispositivo e suas aplicações estão seguros de possíveis ataques sempre precisa de uma etapa fundamental: o teste. O teste de invasão, nesta área chamado de *Pentest*, é a tentativa na prática de se tentar descobrir pontos de vulnerabilidades em seu objeto de estudo, e é parte vital do ciclo de vida de um sistema. A partir dos resultados desses testes é possível tornar o sistema mais robusto, seguro e tolerante a falhas. O conceito é também discutido em detalhes no livro [Moreno 2015].

Este trabalho propõe o estudo e análise de uma aplicação *mobile*, utilizando um celular Android®, para por em pauta alguns padrões e ferramentas normalmente utilizados para atividades de *pentest* e engenharia reversa. Existem diversas vulnerabilidades conhecidas e que recorrentemente são introduzidas em novas aplicações, muitas vezes por desconhecimento do desenvolvedor. A OWASP MAS [MAS 2024] elenca as 10 principais vulnerabilidades encontradas em aplicativos *mobile*. Neste trabalho foi utilizado os padrões e guias fornecidos por eles, por se tratarem de uma grande referência na área de segurança.

Aplicações vulneráveis são alvo de diversos ataques e põem em risco tanto a segurança de seus usuários como também da empresa por trás do software. O melhor entendimento da importância de aplicações seguras e como por isso em prática é crucial para a manutenção e saúde de sistemas [Soussi 2024]. Neste cenário, este trabalho visa mostrar passo a passo os testes realizados em uma aplicação, as vulnerabilidades encontradas, e como conseguir se prevenir.

2. Método de Pesquisa

Este estudo de caso foi conduzido seguindo os padrões propostos pela OWASP MAS, o MASTG (*Mobile Application Security Testing Guide*), neles são propostos 6 enfoques de testes: Armazenamento, Criptografia, Autenticação, Plataforma, Código e Resiliência.

Seguindo a ordem proposta de testes dos enfoques selecionados, por resumir que não foram encontradas muitas informações relevantes nas áreas de Autenticação, Plataforma, Código e Resiliência, já que, numa primeira análise, a aplicação não mostrou nenhuma vulnerabilidade crítica. Como a criptografia se mostrou muito pouco explorada, foi na área de Armazenamento onde foi encontrado com maior facilidade as vulnerabilidades desta aplicação.

Desta forma, ainda seguindo os casos de testes propostos pela OWASP, foram conduzidos diversos testes que serão melhor explicados na sessão de *Estudo de Caso*. Em certos pontos, se fez necessário a utilização de algumas ferramentas de *pentesting* para encontrar as vulnerabilidades desta aplicação, as já consagradas no mundo de *Pentest mobile*, a saber:

- Frida: Descrita como kit de ferramentas para desenvolvedores, engenheiros reversos e pesquisadores de segurança, sendo utilizada a "*frida-server*" para a conexão do *device* de teste com o computador para a injeção de *scripts* [Frida 2024].
- Medusa: Descrito como *framework* extenso e modularizado que automatiza processos e técnicas de análise dinâmicas de aplicações *mobile*. Utiliza o servidor Frida pra fazer a conexão com o *device* [Medusa 2024].

O dispositivo utilizado para se realizar os testes foi um celular Android®, modelo *One Plus One*, com acesso total ao sistema, mais conhecido como celular "rootado" [Okta 2023].

Um celular Android®, por padrão de fábrica, vêm com bloqueios e restrições sobre o quê consegue ser acessado, como por exemplo, no sistema de arquivos não é possível se acessar os arquivos do sistema nem o armazenamento interno das aplicações instaladas.

Para burlar tais restrições, é possível "rootar" [1] um celular, e dessa forma, conseguir acessos irrestritos em um sistema. Existem diversas formas de se conseguir obter esse acesso, que será explicado em mais detalhes no artigo *Smartphone forensic analysis* [Faheem 2024].

Desta forma, como para executar as ferramentas Frida e Medusa, assim como ter acesso à arquivos de dados locais salvos pela aplicação, é necessário permissões de super usuário, justificando a utilização de um celular "rootado" para a realização deste estudo.

3. Estudo de caso

Durante o estudo, foi seguido o padrão OWASP, foi seguido o checklist [MASTG 2024], aplicando os casos de testes estipulados pela norma MASTG e os controles dados pela norma MASVS.

Tais ferramentas permitem investigar como uma aplicação recebe, armazena e processa os dados de clientes. Consequentemente, um agente malicioso externo poderá obter informações de como esta aplicação foi projetada e funciona, o que será o 1º passo. Num 2º passo, aplicará *pentesting* específicos para descobrir as vulnerabilidades existente nesta aplicação.

Este trabalho focará nos testes relacionados ao armazenamento local, isto é, os dados que o aplicativo guarda no celular. O objetivo é **encontrar dados sensíveis que exponha o usuário** do aplicativo. Considerando agentes maliciosos externos, o armazenamento de dados pessoais do usuário pela aplicação de forma insegura representa um grande risco.

O ambiente para execução do *pentesting* pode ser descrito como:

- Um celular Android® com acesso *root* e com a aplicação de teste instalada;
- No celular o servidor Frida rodando;
- Um computador, conectado ao celular via ADB (*Android Debug Bridge*);
- No computador a ferramenta Medusa devidamente configurada com conexão ao celular via servidor Frida.

[1] expressão advinda do usuário *root* ou "super usuário"

Inicialmente, tentou-se rodar a aplicação de maneira direta, sem outras ferramentas. Assim, já foi descoberta a primeira camada de proteção da aplicação, visto que a mesma não estava conseguindo inicializar-se de maneira adequada. Depois de alguns testes foi detectado que ela contava com proteções *anti root*: detectava que o celular estava com acesso *root* e tentava bloquear o acesso.

Outra proteção identificada foi que a aplicação também contava com o uso do Dexprotector [DexProtector 2024], que se trata de uma biblioteca que fornece mais proteção à aplicativos contra possíveis invasões e engenharias reversas.

Neste cenário, o código da aplicação é ofuscada, o que dificulta a introdução de códigos maliciosos nesta aplicação. Mas seria uma nova motivação para trabalhos futuros.

Com estas descobertas, foi necessário encontrar formas de se contornar tais proteções. Para isto foi utilizado alguns *scripts* providos pela ferramenta Medusa. Com isto, a aplicação, e seus mecanismos *anti root*, conseguem ser desativados, sendo possível executar efetivamente uma aplicação sem seus mecanismos de proteção.

Com a aplicação executando, foi possível prosseguir com os testes. Como o aplicativo utilizado requer um usuário autenticado para se acessar suas funcionalidades, o próximo passo foi passar pelo processo de autenticação (cadastro/login). Nesta etapa, verificou-se que são requeridos do usuário inúmeros dados pessoais (sensíveis), situação que foi explorada nos próximos passos do estudo.

Após devidamente configurado e logado no aplicativo, foi iniciado a busca por dados sensíveis expostos pela aplicação. Através do terminal do ADB, como o dispositivo possuía o acesso *root*, foi possível acessar o armazenamento do sistema. Em ambientes Android®, normalmente as aplicações fazem seu armazenamento local dentro da pasta */data/data/<identificador da aplicação>*. Assim que este diretório foi encontrado, todos os dados da aplicação foram extraídos para o computador para uma melhor análise.

Entre as pastas extraídas, os caminhos que são mais comumente utilizados por aplicações *mobile*, e os caminhos oferecidos pelo caso de teste da OWASP, estão dentro das pastas: *databases*, *files* e *shared-prefs*. Nelas são armazenadas as seguintes informações, respectivamente:

- Bases de dados SQLite;
- Quaisquer arquivos criados pela própria aplicação;
- Base de dados *Shared Preferences*, que é um *NoSQL* muito comumente usando em aplicações Android®s já que vem junto do próprio *framework*.

Investigando os arquivos dentro de *files* foram encontradas chaves de autenticação salvas em texto plano, tanto o *access token* como o *refresh token* [Oauth 2024], que podem ser explorados para se realizar requisições com os servidores da aplicação.

Seguindo a análise dos dados tanto salvos em *databases* como *shared-prefs* foram encontrados além dos mesmos *tokens* de autenticação como:

- Dados de usuário como: nome, e-mail, telefone, números de documento e até link para foto de perfil;
- Dados referentes ao funcionamento da aplicação.

4. Trabalhos relacionados

Nesta seção, apresentam-se trabalhos relevantes relacionados à segurança e *Pentest* em aplicativos Android®. A revisão da literatura visa fornecer uma visão abrangente dos estudos existentes, destacando abordagens, metodologias e ferramentas relevantes para uma investigação, a saber:

1. **Android® Application Security: A Survey** [Ghouzali 2024]

(a) Aborda a crítica questão da segurança dos dados armazenados em *smartphones* Android®. No período de 2013 a 2018, os autores exploram ameaças físicas e de software ao modelo de armazenamento do Android®, evidenciando a vulnerabilidade da chave de criptografia, apesar das soluções de criptografia oferecidas. Destaca-se a falta de classificação da segurança no armazenamento, como uma das principais preocupações, conforme indicado pelo projeto de segurança móvel da OWASP. A revisão inclui trabalhos anteriores sobre segurança de *smartphones*, enfatizando a evolução deste tópico, e oferecendo diretrizes para diferentes níveis de segurança. O artigo também fornece uma visão geral do Android®, destacando classes de armazenamento e as medidas de controle de acesso implementadas. Também cita e avalia ameaças físicas (*cold boot, evil maid, RowHammer*) e de software (*malware*, exploração de desenvolvimento deficiente). A revisão contribui para a compreensão aprofundada das questões de segurança no armazenamento de dados Android®, orientando pesquisas futuras e evidenciando a necessidade contínua de aprimoramentos na proteção de dados em dispositivos móveis.

2. **Vulnerability detection in recent Android® apps: An empirical study** [Iqbal 2024]
   (a) O artigo destaca o crescente e rápido aumento no uso de aplicativos para *smartphones*, levando ao armazenamento ampliado de informações sensíveis dos usuários. Essa tendência motiva desenvolvedores maliciosos a explorarem vulnerabilidades em aplicativos de utilidade, buscando capturar dados sensíveis. Devido à natureza mais aberta do Android® e à popularidade entre desenvolvedores amadores, os aplicativos Android® frequentemente se tornam alvos de *malwares*. O estudo empírico realizado investiga uma seleção desses aplicativos, identificando oito vulnerabilidades com a utilização de três ferramentas de qualidade. Os resultados revelam vulnerabilidades nos aplicativos testados, apresentam estatísticas e discutem contramedidas. A pesquisa visa beneficiar os desenvolvedores e, indiretamente, os potenciais usuários desses aplicativos, contribuindo para a segurança e confiabilidade das aplicações.
3. **Smartphone Forensic Analysis: A Case Study for Obtaining Root Access of an Android® Samsung S3 Device and Analyse the Image without an Expensive Commercial Tool** [Faheem 2024]
   (a) O trabalho destaca a importância da análise forense de *smartphones* diante do crescente uso desses dispositivos para comunicação e atividades diversas. O autor enfatiza os riscos de segurança associados ao uso de *smartphones*, considerando a possibilidade de envolvimento em crimes digitais. O estudo de caso apresenta a obtenção de acesso *root* a um dispositivo Samsung S3, a criação de uma imagem DD e a análise dessa imagem sem a necessidade de ferramentas comerciais caras. O autor destaca o desafio para especialistas forenses extrair dados de *smartphones* para fins legais e como o estudo contribui para superar essa dificuldade. O trabalho relacionado destaca uma pesquisa anterior sobre a análise forense de aplicativos de mensagens instantâneas em dispositivos Android®, enfatizando o uso do *UFED Physical Analyzer*. O estudo atual propõe abordar a análise de dados excluídos. O caso de estudo descreve as etapas para obter acesso *root* e criar uma imagem forense do dispositivo.
4. **Health Vulnerabilities in Software Ecosystems: Five Cases of Dying Platforms** [Soussi 2024]
   (a) Neste estudo, a autora explora a dinâmica das ecossistemas de software, definidos como redes de organizações que colaboram para atender a um mercado comum. A saúde de um ecossistema de software é avaliada com base em sua produtividade, robustez e crescimento por meio de parcerias e aquisição de novos membros. Este estudo se concentra em casos de ecossistemas considerados não saudáveis ou em declínio, utilizando análise qualitativa de fóruns comunitários e presença em redes sociais. É

identificado cinco indicadores de decadência de um ecossistema e propõem contramedidas para evitar a sua deterioração. A abordagem do artigo visa auxiliar profissionais a evitar armadilhas que possam prejudicar seus negócios e ecossistemas, oferecendo *insights* estratégicos para tomada de decisões.

Esses trabalhos fornecem uma base sólida para a compreensão das vulnerabilidades em aplicativos Android®, destacando áreas específicas de preocupação e propondo abordagens para mitigar esses riscos.

5. Conclusão e trabalhos futuros

Este estudo de caso permitiu constatar que o aplicativo armazena diversos dados sensíveis, tanto relativos à aplicação como do próprio usuário. Essas informações expostas podem ser exploradas por agentes maliciosos, caracterizando-se como uma grave vulnerabilidade. Uma técnica que poderia ser utilizada para mitigar esse problema seria a criptografia, porém se verificou ser um método muito pouco aplicada, visto a existência de informação criticas não criptografadas.

Como descrito pela própria página da OWASP dados sensíveis armazenados no celular no mínimo deveriam ser encriptados e as chaves de encriptação utilizadas deveriam ser armazenadas utilizando o Android® Keystore, que é uma biblioteca Android® que permite que armazenar chaves criptográficas em um contêiner para dificultar a extração destas informações nos dispositivo.

Como trabalho futuro, a exploração do código ofuscado do aplicativo revela uma nova motivação para novos *pentest* de possíveis vulnerabilidades, seja por engenharia reversa de aplicações, ou por introdução de novos códigos, trazem inúmeras possibilidades de novos testes que podem ser realizados. Um dos possíveis caminhos a ser testado são algumas alterações nos arquivos de configuração da aplicação que poderiam torná-la vulnerável à ataques MITM (*Man in the Middle*).

Referências

[DexProtector 2024] DexProtector (2024). https://licelus.com/products/dexprotector.

[Faheem 2024] Faheem, M. (2024). Smartphone forensic analysis: A case study for obtaining root access of an android samsung s3 device and analyse the image without an expensive commercial tool. https://www.scirp.org/html/2-7800178_47558.htm.

[Frida 2024] Frida (2024). Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers. https://github.com/frida/frida.

[Ghouzali 2024] Ghouzali, H. A. . S. (2024). Android data storage security: A review. https://www.sciencedirect.com/science/article/pii/S1319157818301046.

[Iqbal 2024] Iqbal, F. H. S. . S. F. A. . A. (2024). Vulnerability detection in recent android apps: An empirical study. https://ieeexplore.ieee.org/abstract/document/7885802.

[MAS 2024] MAS, O. (2024). Owasp mas: Define the industry standard for mobile application security. https://mas.owasp.org/.

[MASTG 2024] MASTG (2024). Comprehensive manual for mobile app security testing and reverse engineering. https://mas.owasp.org/MASTG/.

[Medusa 2024] Medusa (2024). Binary instrumentation framework based on frida. https://github.com/Ch0pin/medusa.

[Moreno 2015] Moreno, D. (2015). *Introdução ao Pentest*. Novatec Editora,.

[Oauth 2024] Oauth (2024). Oauth 2.0. https://oauth.net/2/.

[Okta 2023] Okta (2023). Rooted devices: Definition, benefits security risks. https://www.okta.com/identity-101/rooted-device/.

[Soussi 2024] Soussi, L. (2024). Health vulnerabilities in software ecosystems: Five cases of dying platforms. https://studenttheses.uu.nl/bitstream/handle/20.500.12932/30676/Health_Vulnerabilities_Lamia_Soussi_Thesis_Report_FinalVersion.pdf?sequence=2&isAllowed=y.