Plataforma automatizada de intercepción de tráfico en apps móviles con Frida

Fabian A. Gibellini[1], Leonardo R. Ciceri[1], Juliana M. Notreni[1], German N. Parisi[1], Analia L. Ruhl[1], Ninfa M. Zea Cardenas[1], Marcelo J. Auquer[1], Ileana M. Barrionuevo[1], Federico Bertola[1], Sergio R. Quinteros[1] e Ignacio J. Sanchez Balzaretti[1]

[1] Universidad Tecnológica Nacional – Facultad Regional Córdoba
Maestro M. López esq. Cruz Roja Argentina, Ciudad Universitaria, Córdoba, Argentina
{fabiangibellini, leonardorciceri, julinotreni, germannparisi, analialorenaruhl, milyzc, marcelo.auquer, ilebarrionuevo, fedebertola, ser.quinteros, ignaciojsb}@gmail.com

Resumen. En este trabajo se introduce una herramienta unificada que combina diversas técnicas manuales y automatizadas para optimizar la intercepción de tráfico en aplicaciones móviles. Aunque existen soluciones que permiten la captura de tráfico web, muchas presentan limitaciones en términos de cobertura y adaptabilidad a diferentes escenarios de inspección. La herramienta propuesta amplía estas capacidades al integrar Frida y Objection, dos tecnologías que facilitan la manipulación dinámica de objetos y métodos durante la ejecución de la aplicación, ofreciendo así un enfoque más flexible y completo para el análisis del tráfico móvil.

Palabras Claves. Pruebas de penetración, Intercepción, Android, SSL pinning, instrumentación dinámica.

Automated platform for mobile app traffic interception with Frida

Abstract. This paper introduces a unified tool that combines several manual and automated techniques to optimize the web traffic interception process in mobile applications. Although existing solutions allow for web traffic capture, many have limitations in terms of coverage and adaptability to different scenarios. The proposed tool expands these capabilities by integrating Frida and Objection, two technologies that facilitate the dynamic manipulation of objects and methods during application runtime, thus offering a more flexible and comprehensive approach to mobile traffic analysis.

Keywords. Penetration tests, Interceptions, Android, SSL pinning, dynamic instrumentation.

1 Contexto

El análisis dinámico de aplicaciones móviles implica un examen exhaustivo del tráfico de red intercambiado entre el cliente y el servidor, con el objetivo de identificar

vulnerabilidades y entender la lógica operativa de la aplicación. Este proceso presenta algunas complicaciones por la heterogeneidad tecnológica inherente a los entornos móviles, donde coexisten múltiples lenguajes de programación, *frameworks* y plataformas, junto con mecanismos de seguridad diseñados para impedir la interceptación de comunicaciones.

Los profesionales de seguridad (pentesters) enfrentan obstáculos significativos durante esta fase, ya que las herramientas automatizadas disponibles suelen presentar limitaciones. A menudo, requieren configuraciones adicionales o intervenciones manuales prolongadas para lograr una intercepción efectiva. Para abordar esta problemática, la línea de desarrollo en la que se decide hacer foco consiste en la integración de herramientas de instrumentación dinámica en las que la comunidad ha realizado grandes aportes. Lo que se pretende es optimizar el proceso de análisis, reduciendo la carga operativa y aumentando la eficacia en la detección de vulnerabilidades.

2 Introducción

Dentro del espectro de vulnerabilidades en aplicaciones móviles, este trabajo se centra en el ataque de tipo man-in-the-middle (MITM), el cual aprovecha un comportamiento crítico en el protocolo HTTPS: cuando un servidor envía su certificado digital (contiene la clave pública) al cliente, este debe validar su autenticidad. Si el certificado no es verificado adecuadamente, un atacante puede suplantar la identidad del servidor mediante un certificado fraudulento, comprometiendo toda la comunicación (Callegati et al., 2009). Este escenario ocurre cuando los usuarios ignoran las advertencias de seguridad generadas por el navegador o la aplicación.

Este tipo de ataque se enmarca dentro del análisis dinámico activo, donde el evaluador intercepta, modifica y analiza el tráfico de red en tiempo real. Para ello, se emplean herramientas como proxies web (ej: Burp Suite, OWASP ZAP), que actúan como intermediarios, capturando todas las solicitudes y respuestas entre la aplicación móvil y el servidor (Gupta, 2023). Sin embargo, una barrera significativa en este proceso es la implementación de SSL pinning, un mecanismo de seguridad que restringe la aceptación de certificados solo a aquellos predefinidos por la aplicación. Para neutralizar esta protección, el proyecto utiliza técnicas avanzadas con Frida (para instrumentación dinámica) y Objection (especialmente útil en dispositivos sin privilegios de root), permitiendo así la manipulación del tráfico cifrado en entornos restringidos.

3 Líneas de Investigación y Desarrollo

Tras una evaluación exhaustiva de las técnicas y herramientas disponibles, se identificaron las siguientes como fundamentales para el desarrollo del proyecto:

1. **Burp Suite**
    - *Proceso de configuración*:
        - Descarga del certificado CA directamente desde la interfaz de la herramienta.
        - Conversión del formato del certificado para garantizar compatibilidad con dispositivos móviles.
        - Instalación manual en el almacén de certificados del dispositivo objetivo. (PortSwigger, 2024).
2. **Frida**
    - *Requisitos y funcionalidad*:
        - En dispositivos con privilegios de *root*:
            - Instalación de un agente en tiempo de ejecución.
            - Ejecución de scripts personalizados para bypass de SSL pinning.
        - En dispositivos sin *root*:
            - Inyección de un *gadget* en el espacio de memoria de la aplicación objetivo.
        - Compatibilidad multiplataforma (Android/iOS). (Frida, 2024).
3. **Objection**
    - *Metodología de implementación*:
        - Modificación del binario de la aplicación original.
        - Inyección de un *gadget* basado en Frida para instrumentación dinámica.
        - Reempaquetado e instalación de la aplicación modificada.
    - *Ventaja clave*:
        - Soporte para dispositivos tanto *rooteados* como no *rooteados*.
        - Compatibilidad con Android e iOS. (SensePost, 2024).

Existen otras herramientas utilizadas para romper la protección de SSL pinning, como Magisk (Topjohnwu, s.f.), pero requieren que el dispositivo esté rooteado previamente. Se opta por trabajar con las herramientas mencionadas por su capacidad de adaptarse a distintos requerimientos de base del sistema operativo del dispositivo.

Decisión técnica respecto al desarrollo:

Para la automatización de procesos, se optó por desarrollar los scripts en **Python**, debido a:

- Su flexibilidad para integrarse con las herramientas mencionadas.
- Compatibilidad multiplataforma (Windows/Linux/macOS).
- Amplia disponibilidad de bibliotecas para análisis de red y manipulación de datos.

4 Implementación técnica y resultados obtenidos

4.1 Desafíos técnicos y soluciones implementadas

Durante la fase de implementación, se identificaron inconvenientes en entornos Windows debido a configuraciones específicas del *Subsistema de Linux (WSL)*. Para resolverlos, fue necesario ajustar parámetros de red en WSL, permitiendo la integración con la red local de Windows y la correcta identificación de dispositivos móviles conectados. Cabe destacar que tanto **Frida** como **Objection** demostraron compatibilidad tanto con dispositivos físicos como con emuladores, facilitando las pruebas en diversos entornos.

Un avance significativo fue la consolidación de múltiples verificaciones manuales (tipo de dispositivo, arquitectura y estado de *root*) en un **script unificado**, optimizando el proceso de análisis de tráfico. Adicionalmente, se automatizó la instalación de certificados de proxy (inicialmente Burp Suite, pero adaptable a otras herramientas), reduciendo en un **15% el tiempo de configuración manual**.

4.2 Automatización e integración de herramientas

El proyecto incorporó herramientas clave para la interacción con dispositivos móviles, como:

- **ADB** y **Apktool**: Para gestión de dispositivos y manipulación de aplicaciones.
- **Scripts personalizados**: Encargados de deshabilitar *SSL pinning*, ajustar configuraciones de red y modificar archivos críticos en aplicaciones Android.

Todos estos procesos fueron integrados en un flujo automatizado, incluyendo la instalación de dependencias mediante entornos virtuales de **Python**, lo que mitigó conflictos de compatibilidad. La herramienta desarrollada ha demostrado un **ahorro del 70% en tiempo** en la fase inicial del análisis dinámico, durante *penetration tests* realizados por el equipo de pentesters del grupo **GISSIC**. Los criterios de éxito o fracaso de la ejecución, fueron definidos por:

- La generación exitosa e instalación de la apk parcheada mediante objection. Notamos que en la diversidad de sistemas operativos para correr el proyecto, ocurren fallas con frecuencia pero se tuvieron en cuenta las precondiciones para Linux, WSL y MacOS.
- Los pasos adicionales a incorporar cuando se ejecuta con emulador, en donde la intercepción requiere de gestionar con adb ciertas configuraciones extras ya que inicialmente no se lograba la intercepción.
- La detección o no detección de métodos de SSL pinning implementados. El script de Frida utilizado ha logrado cubrir ampliamente diversos mecanismos, por lo que se ha logrado interceptar en todos los casos.

4.3 Discusión

Algunas consideraciones al momento de utilizar la herramienta:

- Objection suele fallar en MacOS con las últimas actualizaciones. Puede ser recomendable mantener una versión menor a la última en general.
- Genymotion sin licencia sólo trabaja con emuladores no rooteados. Esto puede ser un impedimento para el analista en caso de requerir emulador root, por lo cual debe contar con una licencia.
- Genymotion permite trabajar a partir de emuladores con versiones Android 11. Para trabajar con versiones menores, optar por dispositivo físico de la versión requerida, o Android Studio que ofrece mayor cobertura.

4.4 Trabajo futuro

Se planea extender la compatibilidad a dispositivos **iOS**, aprovechando la capacidad nativa de Frida para esta plataforma con el objetivo de un módulo de reconocimiento automático, completando así la cobertura multiplataforma. Actualmente, se está trabajando en la incorporación de *reFlutter* para brindar mayor cobertura en la captura de tráfico de aplicaciones que están desarrolladas con el framework Flutter.

Referencias

Callegati, F., Cerroni, W., & Ramilli, M. (2009). Man-in-the-middle attack to the HTTPS protocol. *IEEE Security & Privacy, 7*(1), 78–81. https://doi.org/10.1109/MSP.2009.12

PortSwigger. (s.f.). *Burp Suite Community Edition*. https://portswigger.net/burp/communitydownload (Último acceso: 19 de junio de 2024)

Frida. (s.f.). *Frida – Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers*. https://frida.re/ (Último acceso: 19 de junio de 2024)

Gupta, A. (2023). *Learning pentesting for Android devices*. Packt Publishing.

SensePost. (s.f.). *Objection – Runtime Mobile Exploration*. https://github.com/sensepost/objection (Último acceso: 19 de junio de 2024)

Topjohnwu. (s.f.). *Magisk – The Magic Mask for Android*. GitHub. https://github.com/topjohnwu/Magisk (Último acceso: 19 de junio de 2024)