Universidad Nacional Autónoma de México

Facultad de Ingeniería
Ingeniería en Computación
Sistemas Operativos

Exposición: Seguridad y privacidad en Linux y XNU en dispositivos móviles

Grupo: 06

Profesor: Gunnar Eyal Wolf Iszaevich

Integrante: 321101464 Michelle Ariana Castañeda González

Introducción

Los sistemas operativos móviles se han convertido en una parte esencial de la vida moderna, al permitir que los dispositivos portátiles funcionen de manera eficiente, segura y conectada. A diferencia de los sistemas de escritorio, los sistemas operativos móviles integran mecanismos de seguridad y privacidad directamente en su diseño, ya que manejan información personal, biométrica y de ubicación de los usuarios. Tanto Android, basado en el kernel Linux, como iOS, basado en el kernel XNU, gestionan memoria, procesos, autenticación y control de hardware para mantener la integridad del sistema y la protección de los datos. Esta investigación analiza cómo ambos sistemas implementan esas medidas de seguridad a nivel interno y por qué son esenciales en el contexto de los dispositivos móviles.

Seguridad y privacidad en Linux y XNU

Procesos e hilos: cómo se aíslan las aplicaciones

La seguridad de un dispositivo móvil depende de la forma en que se gestionan los procesos y los hilos. En Android (Linux), cada aplicación se ejecuta con un identificador único (UID), lo que impide que interactúe con los archivos o la memoria de otra aplicación. Esto se logra mediante la llamada al sistema `clone()`, que crea procesos separados con su propio espacio

de memoria virtual. Como explica Cisco (2021), "cada aplicación se ejecuta en un proceso aislado, con su propio espacio de memoria y permisos asignados por el sistema".

Tanto Android como iOS emplean ASLR (*Address Space Layout Randomization*), una técnica que cambia aleatoriamente la dirección de carga de cada proceso y librería. Esto significa que si una aplicación se ejecuta dos veces, las direcciones de memoria serán distintas en cada ocasión, haciendo ineficaces los ataques basados en ubicaciones fijas. Apple (2024) señala que "ASLR garantiza que cada aplicación se ejecute de forma aislada, minimizando el riesgo de ataques por corrupción de memoria".

En iOS, el kernel XNU refuerza este aislamiento mediante su sistema *sandbox*, que impone reglas estrictas sobre qué archivos o servicios puede usar cada proceso. Si una aplicación intenta acceder a un recurso fuera de su *sandbox*, el sistema la bloquea automáticamente.

Manejo de memoria: cómo se protege la información crítica

La memoria es una de las principales vías de ataque si no se controla correctamente. En los sistemas móviles, el kernel administra la memoria mediante estructuras llamadas páginas, que asignan a cada proceso su propio espacio virtual. Si una aplicación intenta escribir fuera de esa zona, el sistema interrumpe su ejecución (*segmentation fault*), protegiendo el dispositivo de errores o ataques.

Android refuerza esta protección con ASLR y el asignador Scudo, diseñado para detectar comportamientos anómalos en la gestión de memoria (USENIX, 2024). Por su parte, Apple introdujo `kalloc_type` en iOS 15, un sistema que clasifica y protege los bloques de memoria del kernel según su tipo de dato (Apple Security Research, 2025). Estas estrategias previenen ataques de escalada de privilegios, manteniendo protegidos datos sensibles como contraseñas, claves y *tokens* de autenticación.

Autenticación y control de acceso: del usuario al kernel

Desde la perspectiva del usuario, los métodos de autenticación parecen similares en Android e iOS PIN, contraseña, huella digital o reconocimiento facial, pero las diferencias están en cómo se procesan y validan esos datos internamente.

En Android, la autenticación biométrica se gestiona mediante la API `BiometricPrompt`, que envía los datos del sensor al *Trusted Execution Environment* (TEE). Este entorno seguro dentro del procesador valida las huellas o rasgos faciales sin exponerlos al sistema operativo principal (Google, 2024). Además, Android aplica SELinux, un módulo de control obligatorio que restringe qué procesos pueden interactuar con archivos o servicios del sistema.

En iOS, los datos biométricos se procesan en el chip *Secure Enclave*, un subsistema criptográfico aislado del kernel que almacena las claves y realiza las validaciones de identidad (Apple, 2024). Esto significa que ni siquiera el propio sistema operativo puede acceder a los datos biométricos del usuario, reforzando la privacidad a nivel de hardware.

Conclusión

La seguridad y privacidad en los sistemas operativos móviles, tanto en Linux (Android) como en XNU (iOS), se sostienen sobre un enfoque integral que abarca la gestión de procesos, memoria y autenticación desde el núcleo del sistema hasta el hardware. El aislamiento de aplicaciones y el control estricto de los hilos y recursos evitan accesos no autorizados, mientras que técnicas como ASLR y mecanismos de protección de memoria reducen la exposición a ataques de corrupción o escalada de privilegios. Los sistemas de autenticación y control de acceso, respaldados por entornos seguros como TEE o *Secure Enclave*, aseguran que los datos sensibles permanezcan protegidos. Estas estrategias combinadas garantizan que la información crítica del usuario se mantenga segura, estableciendo una base sólida de confianza y privacidad en los dispositivos móviles.

Bibliografía

- Android Developers. (2024). Requesting app permissions. `https://developer.android.com/training/permissions/requesting`
- Android Security. (2024). Android Security Overview. `https://source.android.com/security`
- Apple Inc. (2024). iOS Security Guide. Apple Support. `https://support.apple.com/guide/security/welcome/web`
- Apple Security Research. (2025). Memory Integrity Enforcement in XNU. `https://security.apple.com/blog/memory-integrity-enforcement`
- Cisco Systems. (2021). Mobile Device Security Overview. `https://www.cisco.com/c/en/us/about/security-center.html`
- Google Support. (2024). Trusted Execution Environment on Android. `https://support.google.com/android/answer/15146908`
- USENIX. (2024). Android Memory Protection and Exploit Mitigations. `https://www.usenix.org`