Universidad Nacional de Rosario

Tesina de grado
para la obtención del grado de
Licenciado en Ciencias de la Computación

---

# Hacia un prototipo certificado del sistema de seguridad de Android 9

---


*Alumno:*
Guido De Luca

22 de agosto de 2019

# Índice general

# Estado del arte

En esta sección se describen algunos trabajos sobre el modelo de seguridad de Android. *A priori*, podemos distinguir dos grandes grupos: aquellos que reportan alguna falla puntual del sistema y los que buscan dar una descripción más general del mismo, ya sea de manera formal o informal[1].

## Fallas puntuales

Dentro del primer grupo mencionado podemos encontrar bla bla bla. TODO: Buscar trabajos que apunten a vulnerabilidades específicas. El paper de Bagheri [3] tiene muchas referencias a este tipo de trabajos.

## Enfoque informal

Por otro lado, encontramos trabajos como el de William Enck *et.al* [8], uno de los primeros artículos académicos en describir el modelo de seguridad de Android. A través de esta descripción, los autores buscaban *desenmascarar* la complejidad a la que debían enfrentarse los desarrolladores a la hora de construir aplicaciones seguras.

traducción literal de unmask

Recientemente, René Mayrhofer *et al.* [11] publicaron un trabajo de similar basado en la versión 9.0 de Android. El mismo, además de dar un descripción detallada del modelo, incluye una discusión de sus implicaciones y un posterior análisis sobre las medidas que se tomaron a lo largo del tiempo para mitigar distintas amenazas.

llenar gap temporal con algún trabajo más, puede ser [1]

Este tipo de trabajos constituyen un complemento importante a la documentación oficial de Android, brindándole nuevas herramientas y referencias más claras a los desarrolladores. Un ejemplo de esto fue el trabajo de Felt *et al.* [9], quienes estudiaron un grupo de aplicaciones disponibles para la versión 2.2 de Android y detectaron que muchas de ellas pedían más permisos de los que realmente necesitaban. Los autores investigaron las causas de sobreprivilegio de estas aplicaciones y encontraron que muchas veces, los desarrolladores intentaban otorgar la menor cantidad de privilegios necesarios pero en reiteradas ocasiones fallaban por falta de una documentación precisa. En consecuencia, el grupo desarrolló Stowaway, una herramienta pionera en la detección de permisos innecesarios. En esta línea encontramos también el trabajo de Kathy Wain Yee Au *et al.* [2], el de Piper Chester *et al.* [7], el de Alexandre Bartel *et al.* [5], y el de Sha Wu y Jiajia Liu [14]; siendo este último el más reciente, realizado con la intención de contrarrestar las limitaciones detectadas en los trabajos previos.

en [14] se comparan todos estos trabajos, incorporo esa comparación acá?

[1] Entendemos por enfoque informal a aquellos enfoques que no utilizan métodos formales para el estudio de la plataforma.

### Enfoque formal

Entrando en el terreno de los métodos formales, Chaudhuri [6] presentó un lenguaje tipado que permite describir un subconjunto de aplicaciones de Android y razonar sobre ellas. Adicionalmente, presentó un sistema de tipos para este lenguaje, garantizando que las aplicaciones bien tipadas preservan la confidencialidad de los datos que manejan. Recientemente, Wilayar Khan *et al.* [10] formalizaron y demostraron la corrección de este sistema de tipos utilizando el asistente de pruebas Coq.

Por otra parte, Bagheri *et al.* [4] proponen una formalización del sistema de permisos de Android escrita en Alloy [12], un lenguaje basado en la lógica relacional de primer orden, análisis capaz de realizar *bounded verification* de los modelos que en este lenguaje se describan. Con la ayuda de este modelo, los autores identificaron distintos tipos de vulnerabilidades que permiten esquivar por completo el chequeo de permisos. Particularmente, estudiaron la vulnerabilidad de *permisos personalizados*, mediante la cual una aplicación maliciosa puede acceder a todos los recursos de otra que estén protegidos por permisos personalizados. Esta falla surge de que el sistema no impone restricciones con respecto al nombre de los nuevos permisos que definen y, como consecuencia, dos permisos distintos podrían tener el mismo nombre. Este trabajo luego se extendió para una nueva versión de Android [3]. La falla por permisos personalizados ya había sido reportada por Shin *et al.* en [13].

traducción de bounded verifiaction?

mejor traducción para custom-permission?

debería profundizar más en este párrafo?

Una diferencia fundamental entre este enfoque y el de esta tesina es el tipo de análisis que se realizó. A pesar de que Alloy es capaz de producir contraejemplos de manera automática, algo realmente útil a la hora de buscar potenciales fallas; no es posible demostrar propiedades de una manera rigurosa y formal.

# Bibliografía


[1] Y. Acar y col. "SoK: Lessons Learned from Android Security Research for Appified Software Platforms". En: *2016 IEEE Symposium on Security and Privacy (SP)*. Mayo de 2016, págs. 433-451. DOI: 10.1109/SP.2016.33.

[2] Kathy Wain Yee Au y col. "PScout: Analyzing the Android Permission Specification". En: *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. CCS '12. Raleigh, North Carolina, USA: ACM, 2012, págs. 217-228. ISBN: 978-1-4503-1651-4. DOI: 10.1145/2382196.2382222. URL: http://doi.acm.org/10.1145/2382196.2382222.

[3] H. Bagheri y col. "A formal approach for detection in security flaws in the Android permission system". En: *Formal Aspects of Computing* vol. 30, no. 5 (2018), págs. 525-544.

[4] H. Bagheri y col. "Detection of design flaws in the Android permission protocol through bounded verification". En: *Proceedings of the 2015 International Symposium on Formal Methods* volume 9019 of Lecture Notes in Computer Science (2015), págs. 73-89.

[5] A. Bartel y col. "Static Analysis for Extracting Permission Checks of a Large Scale Framework: The Challenges and Solutions for Analyzing Android". En: *IEEE Transactions on Software Engineering* 40.6 (jun. de 2014), págs. 617-632. ISSN: 0098-5589. DOI: 10.1109/TSE.2014.2322867.

[6] Avik Chaudhuri. "Language-based Security on Android". En: *Proceedings of the ACM SIGPLAN Fourth Workshop on Programming Languages and Analysis for Security*. PLAS '09. Dublin, Ireland: ACM, 2009, págs. 1-7. ISBN: 978-1-60558-645-8. DOI: 10.1145/1554339.1554341. URL: http://doi.acm.org/10.1145/1554339.1554341.

[7] P. Chester y col. "M-Perm: A Lightweight Detector for Android Permission Gaps". En: *2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft)*. Mayo de 2017, págs. 217-218. DOI: 10.1109/MOBILESoft.2017.23.

[8] W. Enck, M. Ongtang y P. McDaniel. "Understanding Android Security". En: *IEEE Security Privacy* 7.1 (ene. de 2009), págs. 50-57. ISSN: 1540-7993. DOI: 10.1109/MSP.2009.26.

[9] Adrienne Porter Felt y col. "Android Permissions Demystified". En: *Proceedings of the 18th ACM Conference on Computer and Communications Security*. CCS '11. Chicago, Illinois, USA: ACM, 2011, págs. 627-638. ISBN: 978-1-4503-0948-6. DOI: 10.1145/2046707.2046779. URL: http://doi.acm.org/10.1145/2046707.2046779.

[10] W. Khan y col. "Formal Analysis of Language-Based Android Security Using Theorem Proving Approach". En: *IEEE Access* 7 (2019), págs. 16550-16560. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2895261.

[11] R. Mayrhofer y col. *The Android Platform Security Model*. 2019. URL: https://arxiv.org/abs/1904.05572.

[12] Software Design Group at MIT. *Alloy*. URL: http://alloytools.org/ (visitado 20-08-2019).

[13] Shin y col. "A small but non-negligible flaw in the Android permission scheme". En: *2010 IEEE International Symposium on Policies for Distributed Systems and Networks* (2010), págs. 107-110.

[14] S. Wu y J. Liu. "Overprivileged Permission Detection for Android Applications". En: *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. Mayo de 2019, págs. 1-6. DOI: 10.1109/ICC.2019.8761572.