Aurasium: Practical Policy Enforcement for Android Applications paper summary

Ruturaj Kiran Vaidya
University of Kansas

1 Summary

The most popular way of securing android device and fighting marware is modifying the operating system to insert monitoring tools to detect the malicious activities. "But, modification of operating system has significant usability issues"[1]. Paper proposes "Aurasium" which provides security and privacy, without any need of os modifications. The idea is - repackage applications to intercept its interactions with the os. There are two main challenges - "adding external code to arbitrary application package and reliably obstructing the interaction with the os" [1]. "Aurasium is made up of two parts: the repackaging mechanism that inserts instrumentation code into Android applications and the monitoring code that intercepts an application's interactions with the os, applying various security policies"[1]. Dex file of an apk is dissembled using apktool and Aurasium native library is added and then dex file is assembled again using apktool. Aurasium wrapped apps include policy logic and the relevant user interface. Alternatively Aurasium Security Manager (ASM) can also be installed, enabling central handling of policy decisions of all repackaged application on the device. The authors also developed a set of security policies, as "Aurasium allows to take full control of the execution of an app"[1]. Authors claim that, Aurasium has shown low overhead and high repackaging success rate making it effective.

2 Contribution

Authors contributes by introducing Aurasium. Aurasium doesn't require any os modifications, rooting of phone, etc. It is applied per app basis. Authors evaluated Aurasium against large number of real world apps and it achieved over 99% of success rate.

3 Strengths

Aurasium shows negligible size overhead and low performance overhead (I won't say 35% is low, but based on the authors' reasoning, I believe that it must be low, or atleast mightn't make a lot of difference). Aurasium doesn't need rooting, flashing or modifications in os. It has shown a pretty high repackaging success rate (99%).

4 Limitations

To me, most important limitations is, it destroys original authors' app signatures (I think the app developer might not allow this). "Also, the existence of Aurasium native library and Java classes allow applications to find out whether it is running under Aurasium or not"[1], and hence I think that Aurasium itself is exploitable. Also, repackaging success rate is 99%, but not 100%.

5 Future Work

Future work is need to protect Aurasium itself! As authors said, it is possible to apply advanced dynamic analyses, eg. information flow and taint analysis. Also, 100% repackaging success is achievable.

References

[1] Rubin Xu and Hassen Saïdi and Ross Anderson *Aurasium: Practical Policy Enforcement for Android Applications*, USENIX Security 12, (2012)