11-4-2019

Automated Dynamic Detection of Self-Hiding Behaviors

Luke Baird
Embry-Riddle Aeronautical University, bairdl1@my.erau.edu

Automated Dynamic Detection of Self-Hiding Behaviors

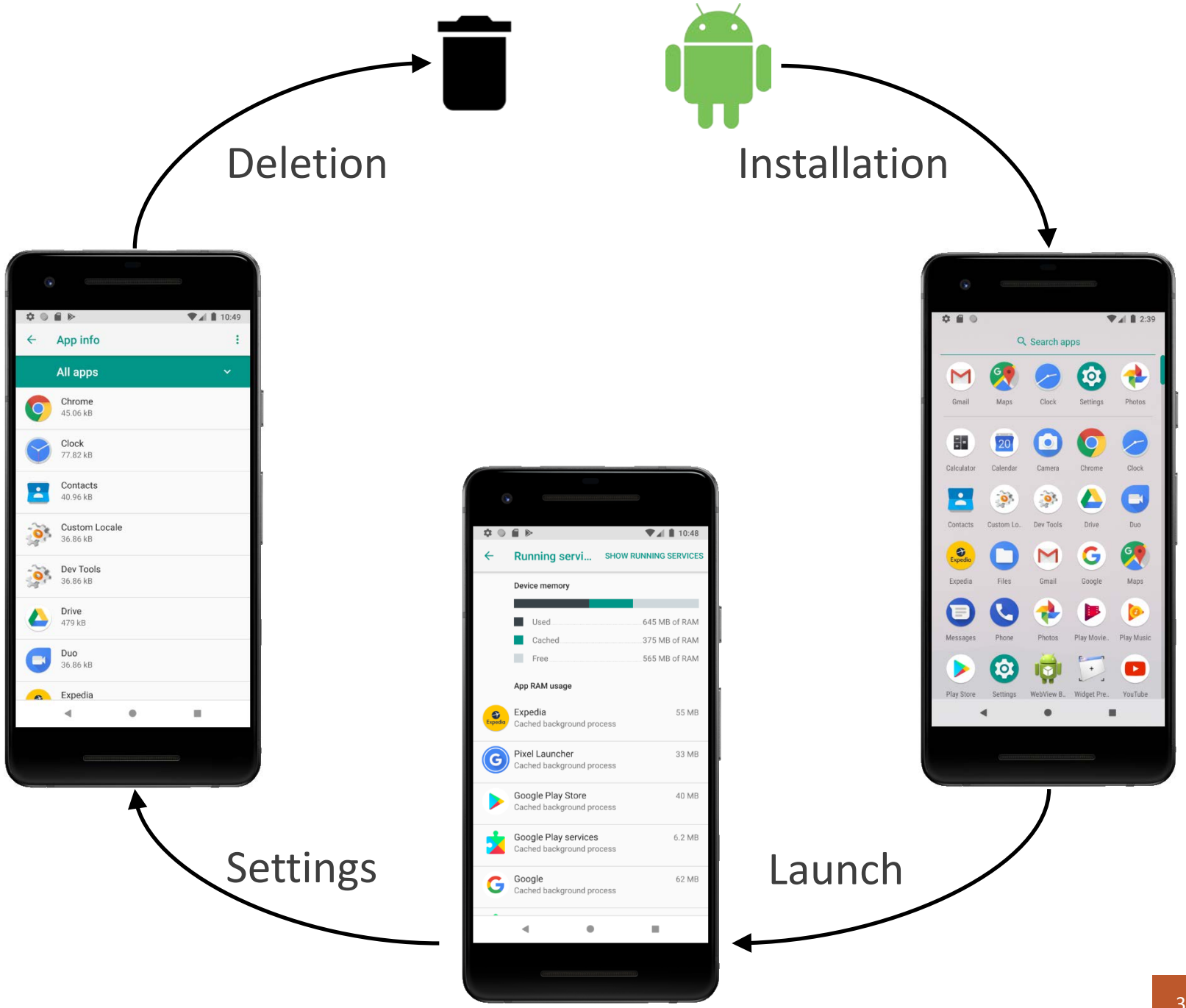
LUKE BAIRD, DR. ZHIYONG SHAN, DR. VINOD NAMBOODIRI

Introduction

What is self-hiding behavior (SHB) in an app?

- *Self-hiding behavior* occurs when an app deliberately conceals itself from the user
- Malicious apps hide from user to avoid detection
- Benign apps may exhibit SHBs
- This introduces a privacy risk to the user

Background—Android App Lifecycle

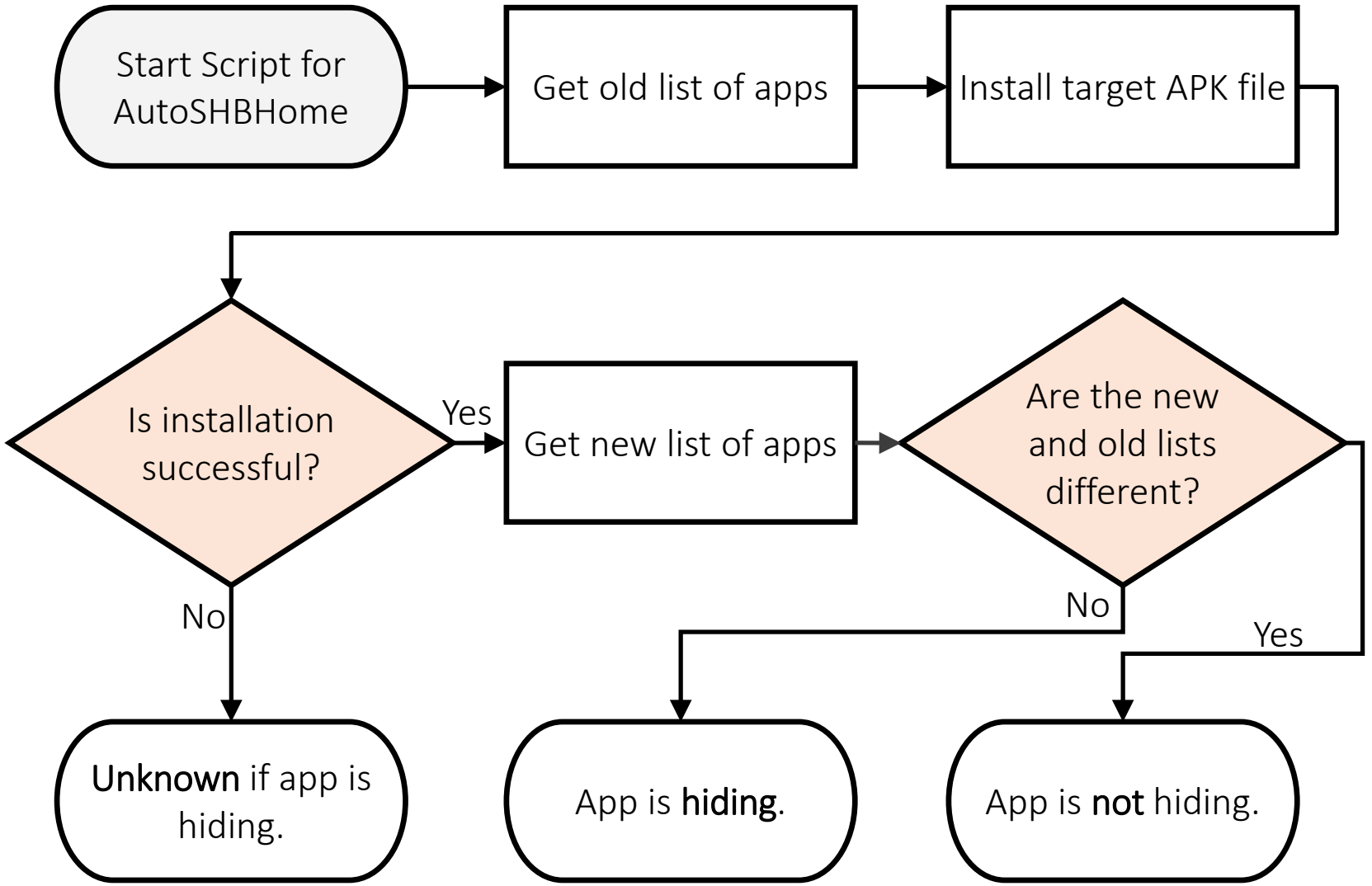


Tool Design

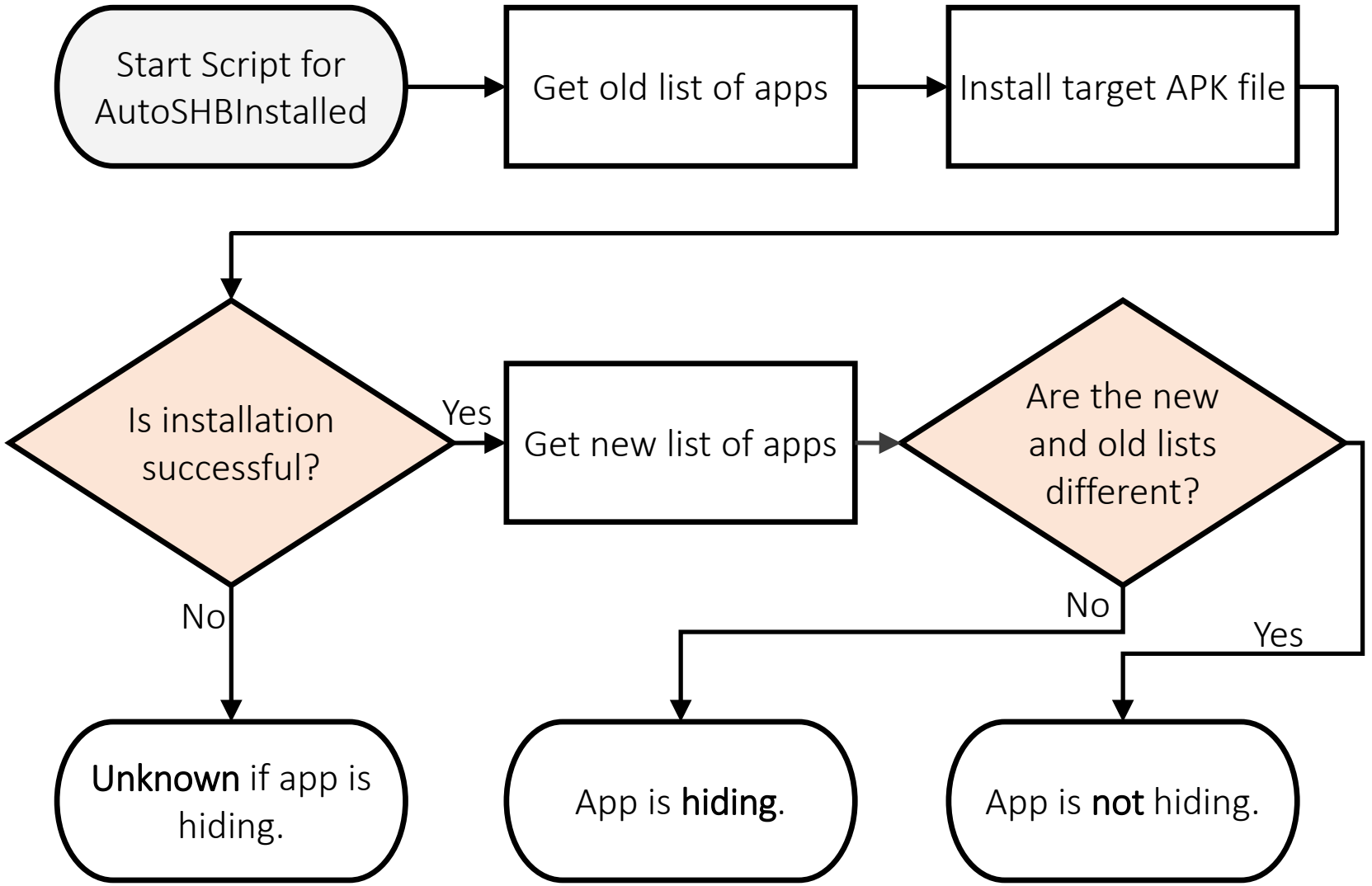
Research developed a set of three tools:

- AutoSHBHome – for detecting SHBs in the home app list
- AutoSHBInstalled – for detecting SHBs in the installed app list
- AutoSHBRunning – for detecting SHBs in the running app list

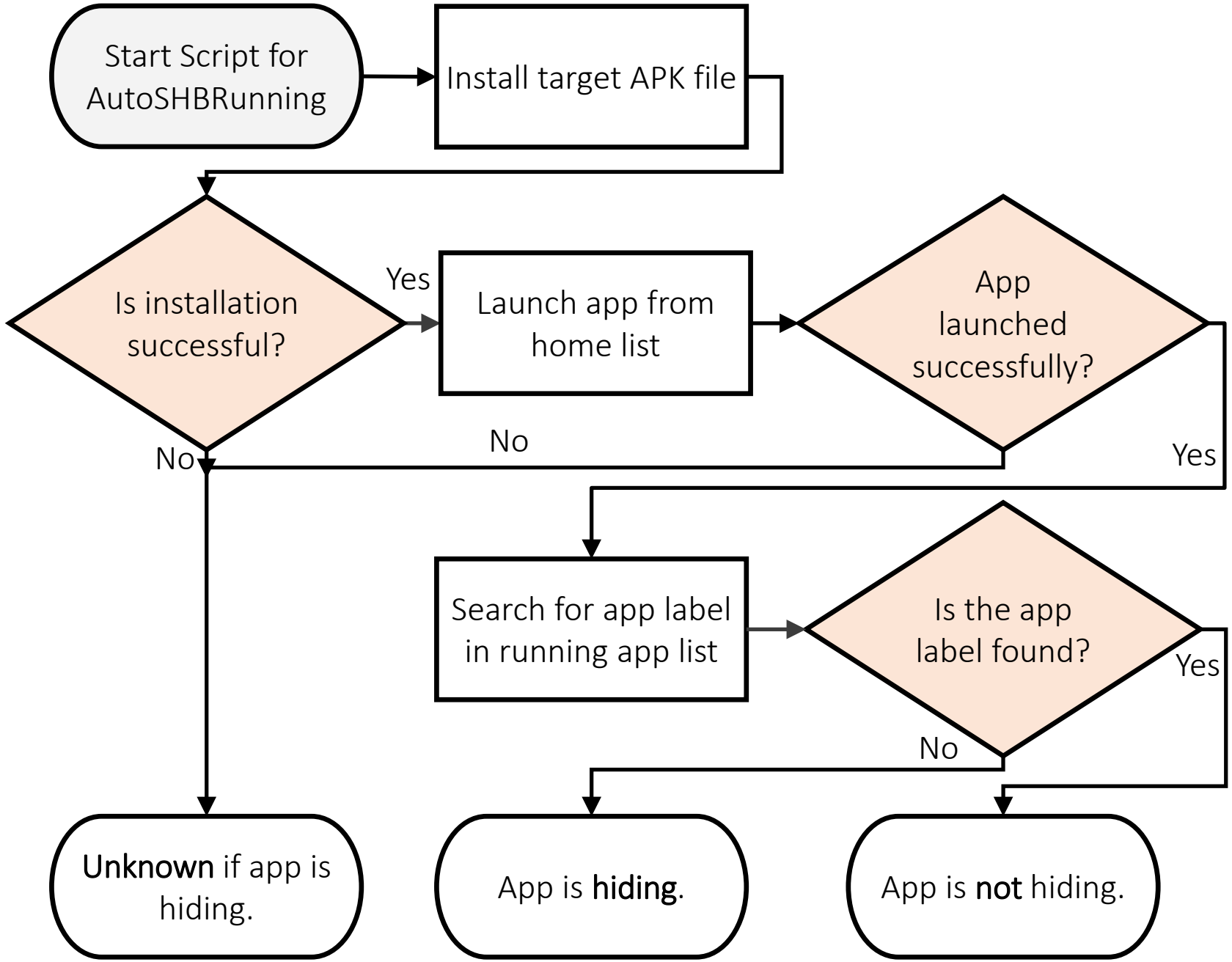
AutoSHBHome Algorithm



AutoSHBInstalled Algorithm



AutoSHBRunning Algorithm



Evaulation – Accuracy of Analysis

Test	# analyzed	# hiding	# not hiding	False positives	False Negatives	Precision	Recall	F-Measure	Errors
AutoSHB Home	77	12	62	2	0	97.47%	100%	98.72%	3
AutoSHB Installed	77	0	76	0	0	100%	N/A	100%	1
AutoSHB Running	63	3	40	0	0	100%	100%	100%	20

Evaluation – Efficiency of Analysis

Test	Total Time	Average time per app	Median time per app	Maximum time per app	Minimum time per app
AutoSHBHome	8569 sec (2.4 h)	85.9 sec	84 sec	139 sec	30 sec
AutoSHBInstalled	14712 sec (4.1 h)	149.3 sec	156 sec	188 sec	76 sec
AutoSHBRunning	10982 sec (3.0 h)	111 sec	96 sec	1373 sec	5 sec

Evaluation – Limitations

- Highly UI-dependent
- Only implemented for Android
- Apps only analyzed if compilation for x86 available

Conclusions & Future Work

Conclusions

- Tools demonstrate effective and efficient technique for finding SHBs

Future Work

- Develop tools for iOS platform
- Research methods to make implementation less UI-dependent

Acknowledgements

- NSF Grant #1659396
- Wichita State University for hosting REU site
- Embry-Riddle Aeronautical University for Travel Funding

Questions?

LUKE BAIRD: BAIRDL1@MY.ERAU.EDU

ZHIYONG SHAN: ZHIYONG.SHAN@WICHITA.EDU

Backup Slides

Script Interfacing

Script

Rest API

Driver
(UiAutomator2)

Interaction

Android Debug Bridge

Home Application Appium Parameters

Desired capabilities JSON parameters:

- platformName: Android
- platformVersion: 8.1
- automationName: uiautomator2
- deviceName: Android Emulator
- appPackage: com.google.android.apps.nexuslauncher
- appActivity: .NexusLauncherActivity

Settings Application Appium Parameters

Desired capabilities JSON parameters:

- platformName: Android
- platformVersion: 8.1
- automationName: uiautomator2
- deviceName: Android Emulator
- appPackage: com.android.settings
- appActivity: .Settings

References

[1] “Adware plagues google play store,” Apr 2019. [Online]. Available: https://blog.avast.com/adware-plagues-google-play

[2] Z. Shan, I. Neamtiu, and R. Samuel, “Self-hiding behavior in android apps: detection and characterization,” in 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE). IEEE, 2018, pp. 728–739.

[3] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, K. Rieck, and C. Siemens, “Drebin: Effective and explainable detection of android malware in your pocket.” in Ndss, vol. 14, 2014, pp. 23–26.

[4] G. Shah, P. Shah, and R. Muchhala, “Software testing automation using appium,” International Journal of Current Engineering and Technology, vol. 4, no. 5, pp. 3528–3531, 2014.

[5] S. Singh, R. Gadgil, and A. Chudgor, “Automated testing of mobile applications using scripting technique: A study on appium,” International Journal of Current Engineering and Technology (IJCET), vol. 4, no. 5, pp. 3627–3630, 2014.

[6] W. Enck, M. Ongtang, and P. McDaniel, “On lightweight mobile phone application certification,” in Proceedings of the 16th ACM conference on Computer and communications security. ACM, 2009, pp. 235–245.

[7] Y. Zhou and X. Jiang, “Dissecting android malware: Characterization and evolution,” in 2012 IEEE symposium on security and privacy. IEEE, 2012, pp. 95–109.

[8] M. Grace, Y. Zhou, Q. Zhang, S. Zou, and X. Jiang, “Riskranker: scalable and accurate zero-day android malware detection,” in Proceedings of the 10th international conference on Mobile systems, applications, and services. ACM, 2012, pp. 281–294.

[9] J.-F. Lalande and S. Wendzel, “Hiding privacy leaks in android applications using low-attention raising covert channels,” in 2013 International Conference on Availability, Reliability and Security. IEEE, 2013, pp. 701–710.

[10] W. Enck, P. Gilbert, S. Han, V. Tendulkar, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, “Taintdroid: an informationflow tracking system for realtime privacy monitoring on smartphones,” ACM Transactions on Computer Systems (TOCS), vol. 32, no. 2, p. 5, 2014.

[11] Z. Aung and W. Zaw, “Permission-based android malware detection,” International Journal of Scientific & Technology Research, vol. 2, no. 3, pp. 228–234, 2013