International Journal of Computing and Business Research (IJCBR)
ISSN (Online) : 2229-6166
International Manuscript ID : 22296166V8I4112018
Volume 8 Issue 4 November 2018

Implementation Patterns of Smartphone Malware Prediction using Deep Learning

Anil Kumar
Assistant Professor
Department of Computer Science
Govt. College for Women, Hisar, Haryana, India

Abstract

Because of the substantial increase in smartphone app usage and Android phone use by mobile customers, the security issues are major. Security vulnerabilities must be addressed to help avoid problems and detect them early. In order to give individuals a heads-up about potential dangers, this technique is connecting smartphone users with one other. People installing APK files don't have to worry about repercussions, such as penalties, since it's common for mobile users to install them from places other than Google Play. Building and implementing a strategy for anticipating malicious Android APKs is critical and hereby integrated for analytics patterns. For our project, we will use the Android APK datasets to begin. Even malignant and benign APKs will be sent. The purpose of this research is to extract signatures that are embedded within the APKs, thus making it feasible to create a training dataset. Around half of the APK files will be safe, while the other half will be malicious. And, after that, we examine permissions in each APK to see what they affect. A dataset is prepared for training the model in order to perform prediction by cleaning it up. Random APKs are chosen to conduct predictive analytics since a vast number of them are required. It is thus possible to discern the likelihood of the presence of

malicious code in the newly analysed APK. Various prediction metrics are measured for both time and cost, utilising machine learning to monitor outcomes. We incorporate our forecasts with machine learning for comparative studies.

Keywords: Android App Security, APK Malware Prediction, Mobile App Security, SmartPhone App Security

Introduction

The enormous research analytics identified a virus that mimics an update mechanism and is thus difficult to locate. Installed, it controls Android phones, where it has the ability to steal text messages, data, and photos, etc. The researchers found that hackers are able to get control of a victim's device, and even capture conversations, images, and texts [1]. Hackers may learn the browser details of the target and take their search history and bookmarks. They are able to view the data that is being copied to the clipboard, as well as glean details about the user's device [2-4].

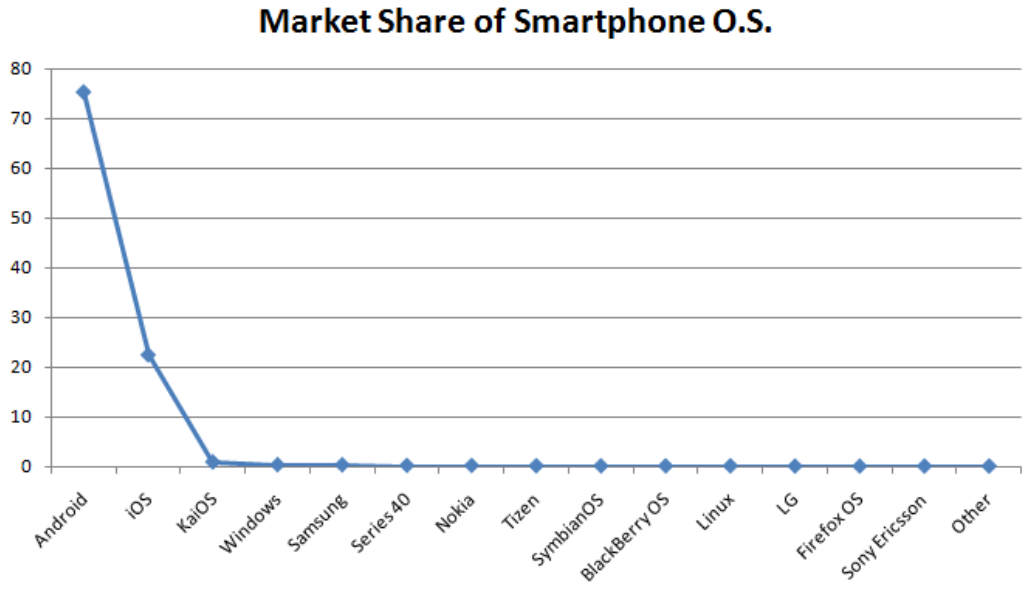

Figure 1: Smartphone Marketshare

Android's package format, called APK, is a file type which is used by the OS to install and distribute mobile apps. The extension .apk, often generated by the Android SDK, designates application files on Android [5-11].

Table 1: Traditional structure of Android APK

APK File / Folder	Description
classes.dex	Compiled Code of Application
AndroidManifest.xml	XML format Manifest File
assets/	Optional Folder for AssetManager
res/	Resources without compilation
META-INF/	Information of Metadata
resources.arsc	Application resources
lib/	Optional Folder with Compiled Code

Permissions in APK

Malware, in a variety of shapes and sizes, is out there to wreak havoc on a system and accomplish a number of evil objectives, including damaging the system, making money, or entering the system illegitimately and leaving security weak or leaking sensitive data [8]. The variety of ways that people can spread viruses is endless, ranging from Trojans to rootkits, and even sending suspicious packers.

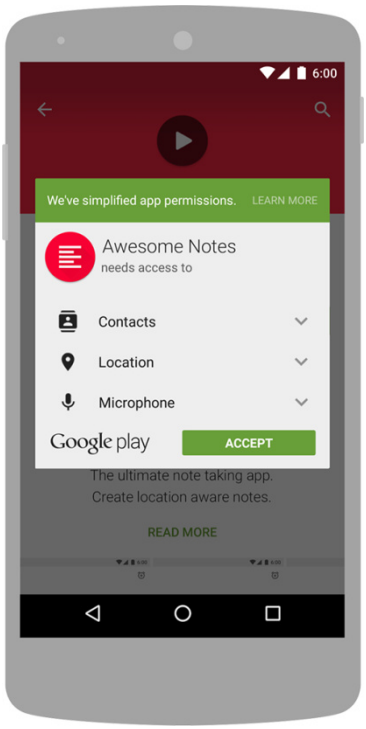

Figure 2 : Permissions in APK

- getpackagesize
- requestdeletepackages
- requestcompanionusedatainbackground
- killbackgroundprocesses
- transmitir
- setalarm
- changewifimulticaststate
- foregroundservice
- setwallpaperhints
- modifyaudiosettings
- readsyncsettings

- changenetworkstate
- accessnetworkstate
- vibrate
- disablekeyguard
- manageowncalls
- expandstatusbar
- requestcompanionruninbackground
- installshortcut
- usefingerprint
- changewifistate
- accesslocationextracommands
- setwallpaper
- requestignorebatteryoptimizations
- readsyncstats
- broadcaststicky
- bluetooth
- writesyncsettings
- reordertasks
- wakelock
- nfc
- accesswifistate
- accessnotificationpolicy
- internet
- bluetoothadmin
- receivebootcompleted

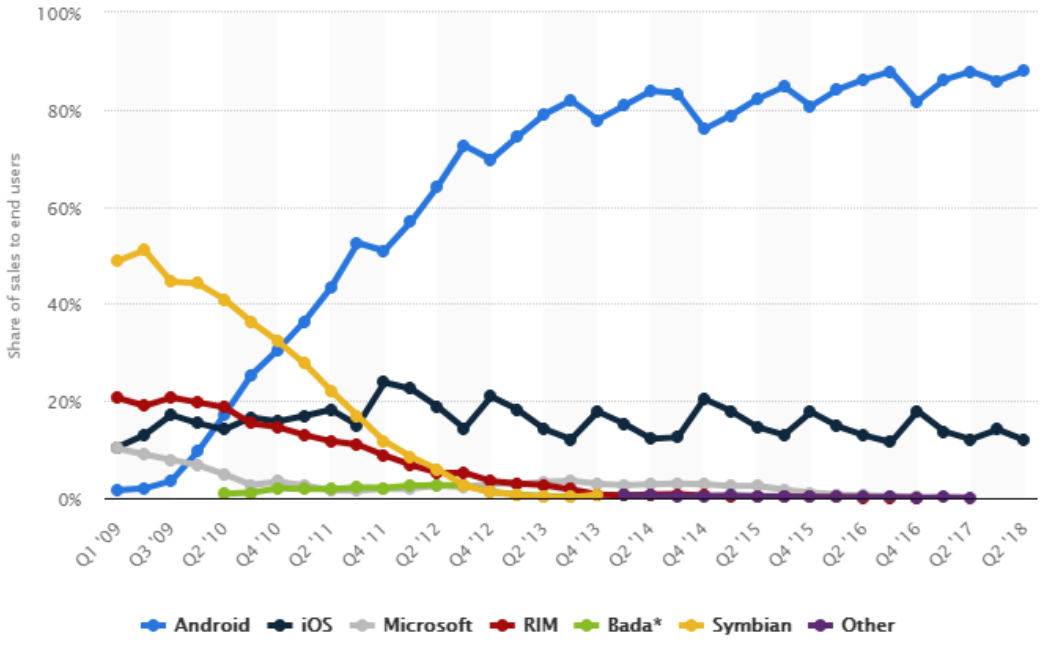

Figure 3 : Market Span of Smartphone O.S.

People have been known to utilise scareware, evasive techniques, backdoors, keyloggers, Trojan game thieves, browser hijackers, ransomware, rogue software, botnets, and even exploit code to cause harm. Two methods malware may propagate are described. Polymorphic malware uses fresh code each time it replicates, and each new version of it seems to be different since it uses its original code. A lot of IDS programmes are available, and there are also free solutions which allow categorization of attacks (by utilising PCAP Files) and can analyse network data [12-16].

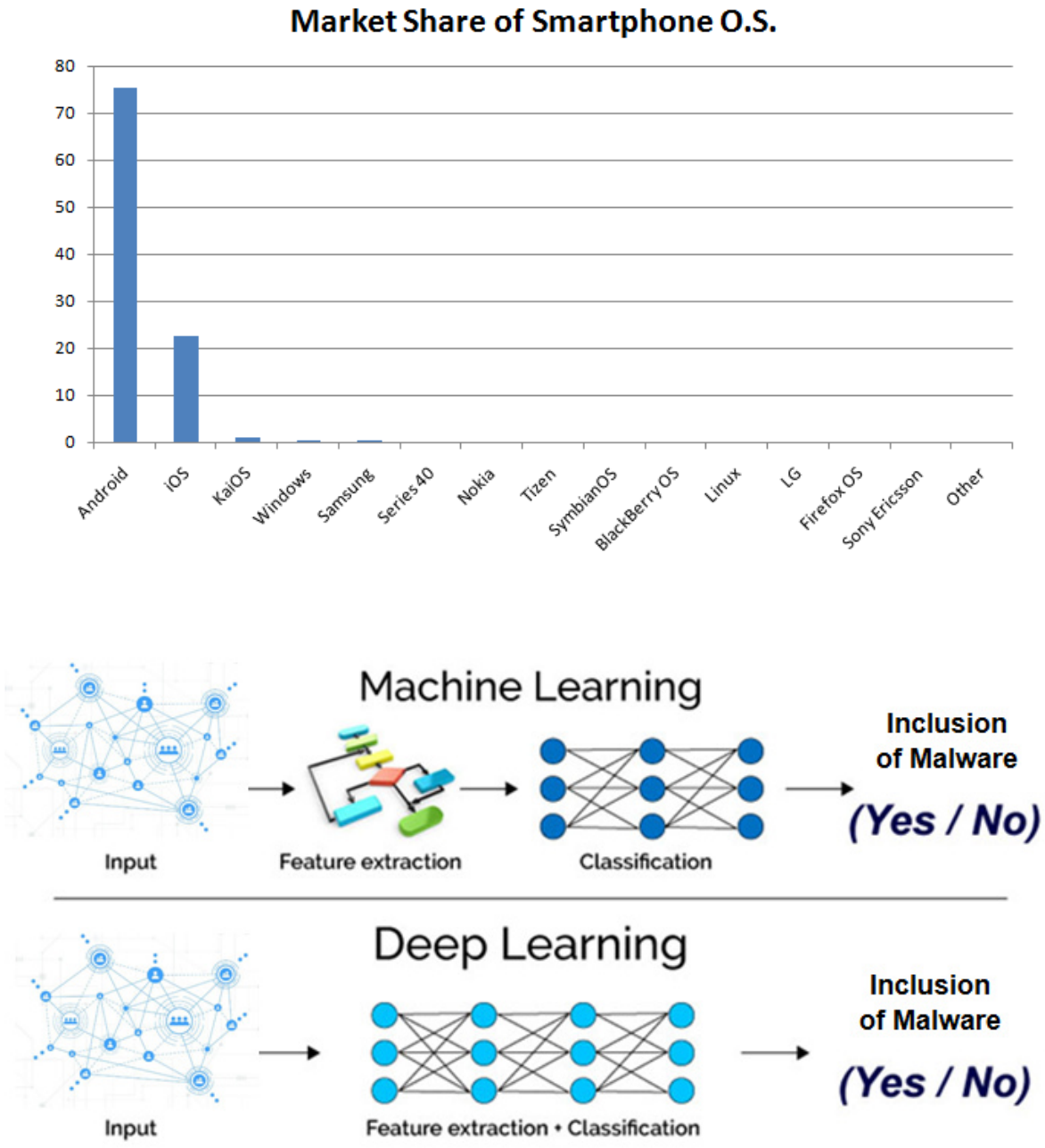

Figure 4 : Machine Learning and Deep Learning

While the risks are rising in the Android ecosystem, users must get the apps they need via a huge number of places. To identify the APK vulnerabilities, you'll want to show the model and method you have available [17].

- comdbirddskydwhaledcamera – Whale Camera
- comdfilterdsweetdcamera – Sweet Camera
- comdroomdwowdcamera – Wow Camera-Beautyï¼ŒCollageï¼ŒEdit
- comdgroupdhotcamera – Hot Camera
- comdblueddeepdcleaner – Deep Cleaner
- comdwalldgooddcleverdcamera – Clever Camera
- globaldfmdfilesexplorer – file explorer
- comdwinddcocodcamera – Coco Camera
- comdwalldfastdcleaner – Fast Cleaner
- comdwalldgooddclevercamera – Clever Camera
- comdwithdswandcamera – Swan Camera
- comdcmdhiporn – HiPorn
- comdogteamdelephantadalbum – Elephant Album
- comdmoredlightdvpn – Light VPN-Fast, Safe,Free
- comdbestdshelldcamera – Shell Camera
- comdstartdsuperdspeedtest – comdqtidatfwddcore
- comdwelldhotdcleaner – Hot Cleaner

Analytics Patterns and Outcomes

International Journal of Computing and Business Research (IJCBR)
ISSN (Online) : 2229-6166
International Manuscript ID : 22296166V8I4112018
Volume 8 Issue 4 November 2018

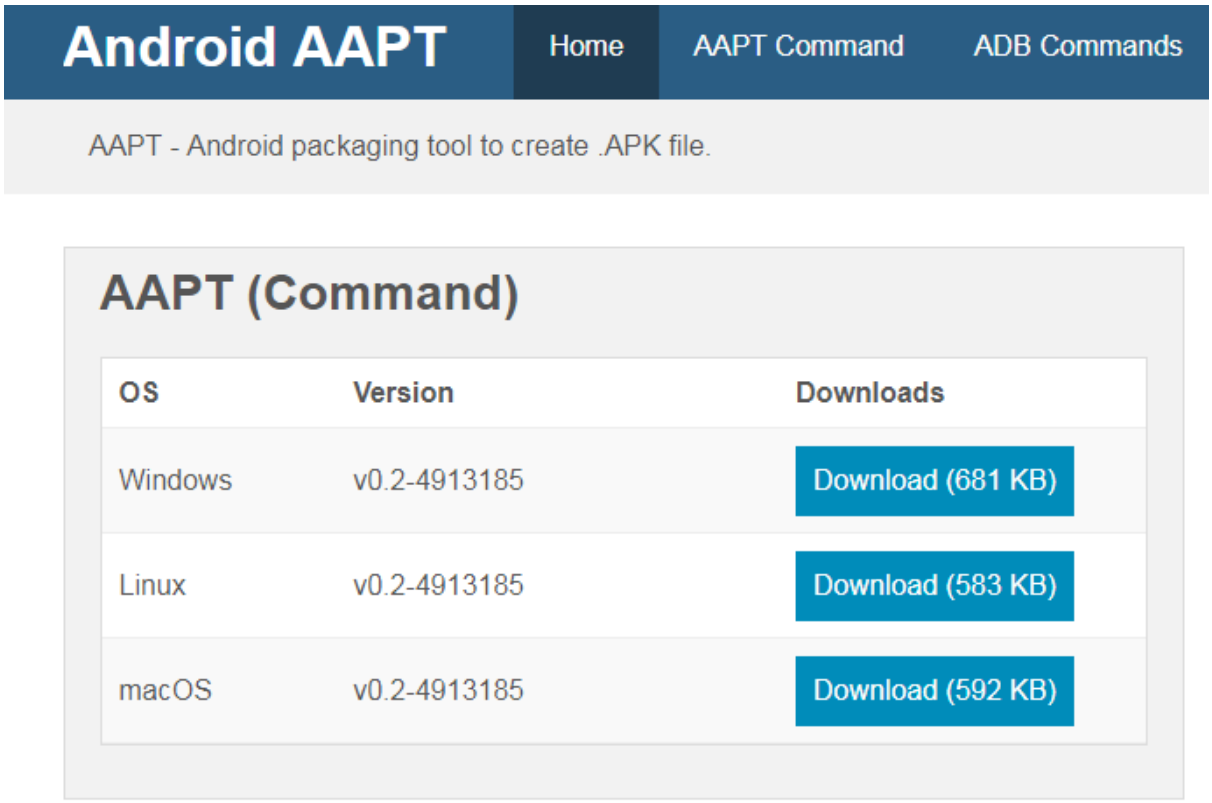

Figure 5 : AAPT Tool

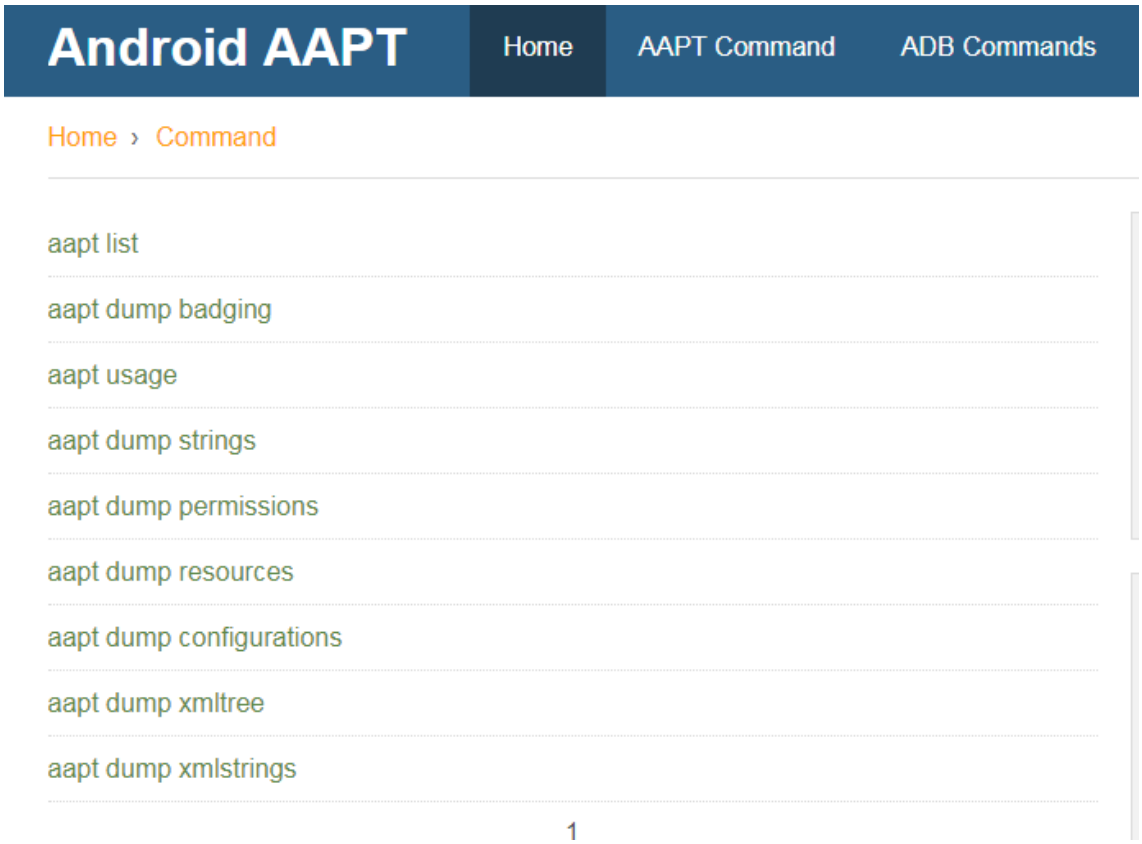

Figure 6 : AAPT Tool

Copy the downloaded APK to AAPT Folder for fetching Permissions

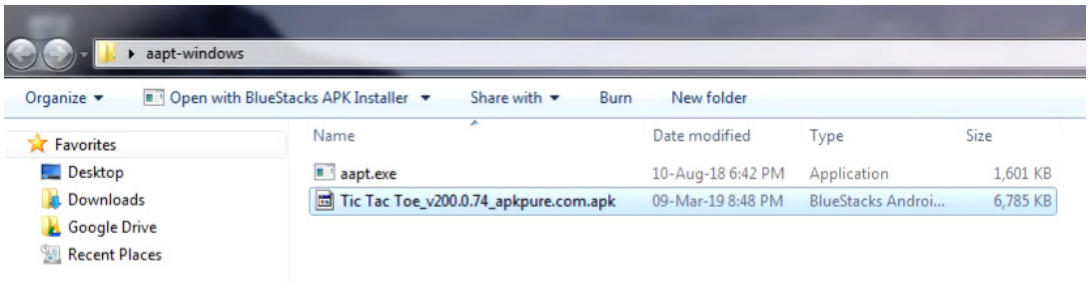

Figure 7 : AAPT Directory

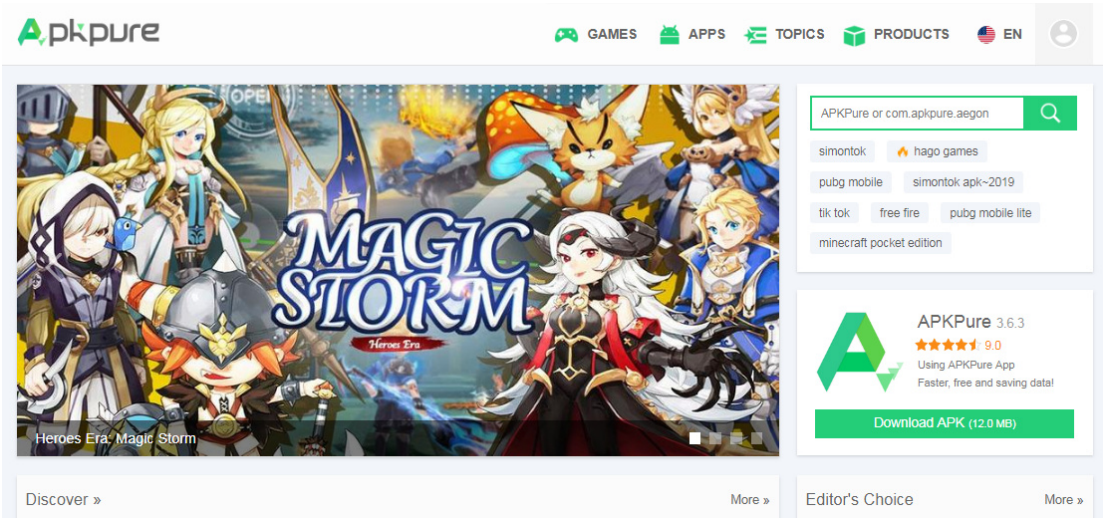

Figure 8 : The Repository of APKPure

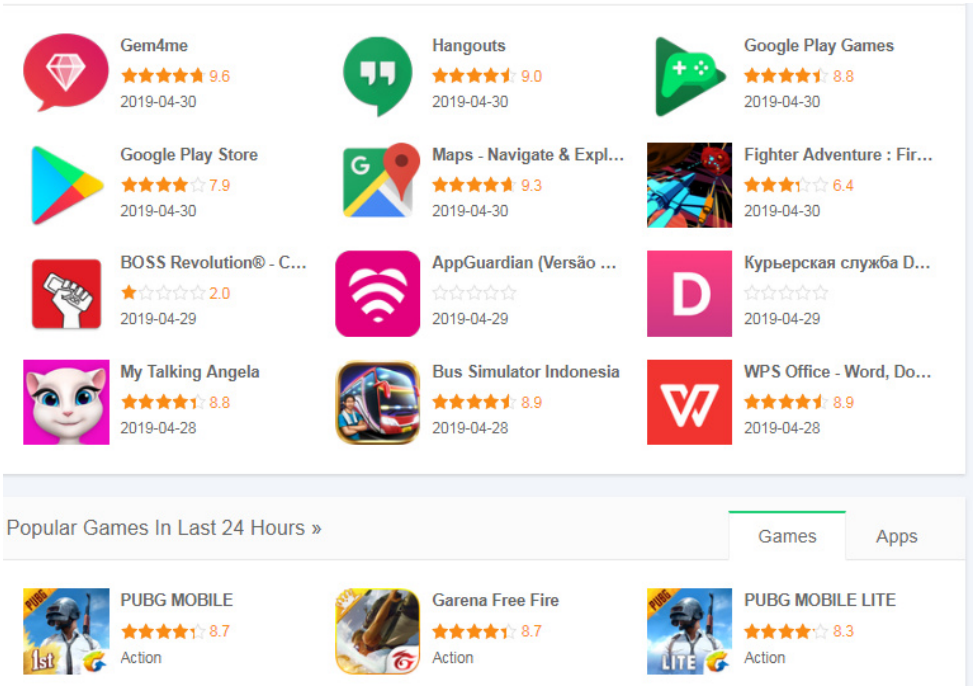

Figure 9 : Apps in APKPure Repository

Table 2 : Formation of Dataset for Implementations

Call	Messaging	Suspicious (0-Malignant, 1-Benign)	Camera	Call	Network	NFC
1	0	**1**	1	0	1	1
0	0	**0**	0	0	0	1
1	1	**0**	1	0	0	0
0	0	**0**	0	0	0	0
0	0	**1**	0	0	0	0
1	0	**1**	1	1	0	0
0	1	**1**	1	1	1	0
0	1	**1**	0	1	1	0
0	0	**0**	0	1	1	0
0	0	**0**	0	0	0	1
1	1	**0**	0	0	0	0
0	0	**0**	0	0	0	0
0	1	**0**	0	1	0	0
0	0	**0**	1	0	0	0
0	1	**0**	0	1	0	0
0	1	**0**	1	0	0	1
0	1	**0**	0	0	1	1
1	1	**0**	0	1	0	1
1	0	**0**	0	0	0	0
1	0	**1**	0	1	0	0

0	1	**1**	1	0	0	0
0	0	**0**	1	0	0	0
0	0	**0**	1	1	1	0
0	0	**0**	0	0	1	0

Table 3 : Evaluation of Parameters

Execution Scenario	**Traditional Machine Learning Approach (Accuracy)**	**Deep Learning Based Approach (Accuracy)**
1	84	94
2	85	95
3	78	98
4	82	94
5	85	96

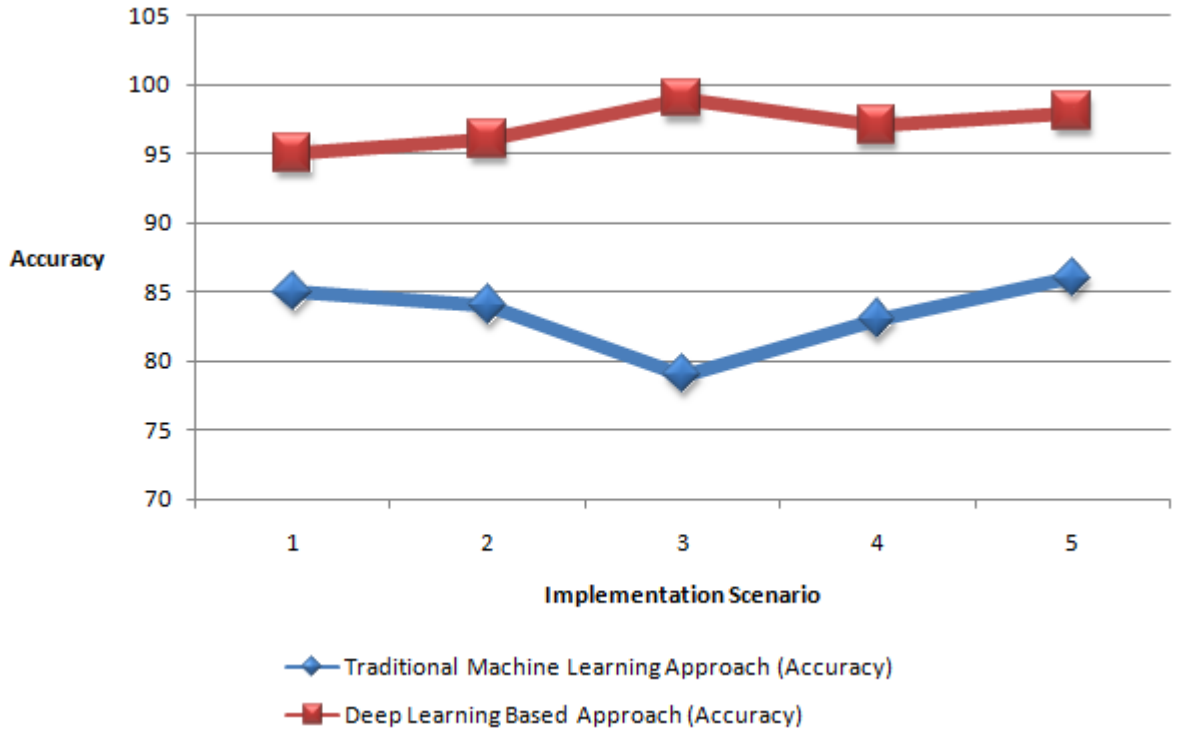

Figure 10 : Evaluation of Execution Time

On the criterion of execution time, deep learning and conventional machine learning were benchmarked to assess their comparative performance. The time to carry out RF in all five tries is discovered to be lower than that of conventional machine learning. This indicates that RF is better optimised than traditional machine learning.

Conclusion

This research focuses on the improved malware detection and predictions using deep learning and their implementation. After looking at Android APKs, researchers found that when classifying the file, a high degree of confidence may be attained by looking at file footprints and signatures. This guide outlines how to successfully do each of the complex tasks while covering

all of the many elements and effectively communicating each step. Developers may use blockchain technology in Android APKs for increased security and privacy. In the area of Cryptocurrency, where study has shown that Blockchain Study and Implementation are very significant. Despite controversy and failure, a range of digital currencies are successful and have gained immense global interest. A myriad of currencies are dubbed "cryptocurrencies," including BitCoin, Ethereum, LiteCoin, PeerCoin, GridCoin, PrimeCoin, Ripple, Nxt, DogeCoin, NameCoin, AuroraCoin, Dash, Neo, NEM, PotCoin, TitCoin, Verge, Stellar, VertCoin, Tether, Zcash, and many more. Because blockchains underlie these cryptocurrencies, transaction data aren't recorded via an intermediate bank or payment provider. The main reason for this prohibition is due to the fact that countries find cryptocurrencies problematic. While many cryptocurrencies, including well-known ones, are strong in terms of security, they still depend on blockchain. Transactions may be fully encrypted and controlled using dynamic cryptography, thus a blockchain is almost invulnerable to hacker and sniffer attempts.

References

[1] Zhou, Y., & Jiang, X. (2012, May). Dissecting android malware: Characterization and evolution. In 2012 IEEE symposium on security and privacy (pp. 95-109). IEEE.

[2] Arp, D., Spreitzenbarth, M., Hubner, M., Gascon, H., Rieck, K., & Siemens, C. E. R. T. (2014, February). Drebin: Effective and explainable detection of android malware in your pocket. In Ndss (Vol. 14, pp. 23-26).

[3] Grace, M., Zhou, Y., Zhang, Q., Zou, S., & Jiang, X. (2012, June). Riskranker: scalable and accurate zero-day android malware detection. In Proceedings of the 10th international conference on Mobile systems, applications, and services (pp. 281-294). ACM.

[4] Yan, L. K., & Yin, H. (2012). DroidScope: Seamlessly Reconstructing the {OS} and Dalvik Semantic Views for Dynamic Android Malware Analysis. In Presented as part of the 21st {USENIX} Security Symposium ({USENIX} Security 12) (pp. 569-584).

[5] Wu, D. J., Mao, C. H., Wei, T. E., Lee, H. M., & Wu, K. P. (2012, August). Droidmat: Android malware detection through manifest and api calls tracing. In 2012 Seventh Asia Joint Conference on Information Security (pp. 62-69). IEEE.

[6] Isohara, T., Takemori, K., & Kubota, A. (2011, December). Kernel-based behavior analysis for android malware detection. In 2011 Seventh International Conference on Computational Intelligence and Security (pp. 1011-1015). IEEE.

[7] Isohara, T., Takemori, K., & Kubota, A. (2011, December). Kernel-based behavior analysis for android malware detection. In 2011 Seventh International Conference on Computational Intelligence and Security (pp. 1011-1015). IEEE.

[8] Aung, Z., & Zaw, W. (2013). Permission-based android malware detection. International Journal of Scientific & Technology Research, 2(3), 228-234.

[9] Sahs, J., & Khan, L. (2012, August). A machine learning approach to android malware detection. In 2012 European Intelligence and Security Informatics Conference (pp. 141-147). IEEE.

[10] Tam, K., Khan, S. J., Fattori, A., & Cavallaro, L. (2015, February). CopperDroid: Automatic Reconstruction of Android Malware Behaviors. In Ndss.

[11] Feng, Y., Anand, S., Dillig, I., & Aiken, A. (2014, November). Apposcopy: Semantics-based detection of android malware through static analysis. In Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (pp. 576-587). ACM.

[12] Yuan, Z., Lu, Y., Wang, Z., & Xue, Y. (2014, August). Droid-sec: deep learning in android malware detection. In ACM SIGCOMM Computer Communication Review (Vol. 44, No. 4, pp. 371-372). ACM.

[13] Zhang, M., Duan, Y., Yin, H., & Zhao, Z. (2014, November). Semantics-aware android malware classification using weighted contextual api dependency graphs. In Proceedings of the 2014 ACM SIGSAC conference on computer and communications security (pp. 1105-1116). ACM.

[14] Yerima, S. Y., Sezer, S., McWilliams, G., & Muttik, I. (2013, March). A new android malware detection approach using bayesian classification. In 2013 IEEE 27th international conference on advanced information networking and applications (AINA) (pp. 121-128). IEEE.

[15] Reina, A., Fattori, A., & Cavallaro, L. (2013). A system call-centric analysis and stimulation technique to automatically reconstruct android malware behaviors. EuroSec, April.

[16] Peiravian, N., & Zhu, X. (2013, November). Machine learning for android malware detection using permission and api calls. In 2013 IEEE 25th international conference on tools with artificial intelligence (pp. 300-305). IEEE.

[17] Petsas, T., Voyatzis, G., Athanasopoulos, E., Polychronakis, M., & Ioannidis, S. (2014, April). Rage against the virtual machine: hindering dynamic analysis of android malware. In Proceedings of the Seventh European Workshop on System Security (p. 5). ACM.