Behavioural Characterization of Android Malware to Detect Similar Malware

Sangeeta Rani[1], Kanwalvir Singh Dhindsa[2]
[1]Research Scholar, [2]Professor
[1]I.K Gujral Punjab Technical University, Jalandhar, Punjab, India
[2]Baba Banda Singh Bahadur Engineering College, Fatehgarh Sahib, Punjab, India

Abstract **-** The pervasiveness of Android smartphones has also accelerated the growth of mobile malware. On the other hand, the current defense mechanisms are affected by the incomplete understanding of upcoming mobile malware families and the dearth of detailed analysis of these malware samples. The contributions of this paper are twofold. First, the research study accomplish the requirements by presenting the collection of 100 Android malware samples in 33 different malware families, that dominate the existing Android malware, in the year 2016. With the help of static and dynamic malware detection techniques, the characterization of collected malware samples based on their detailed behaviour is presented. Second, this paper examines Android application security by analyzing 1946 popular free Android applications. Results reveal that the common characteristics of malware family help in the discovery of similar known and unknown malware.

Keywords **-** Android, Malware, Applications, Permission, Anti-malware software

I. INTRODUCTION

Android is a platform for mobile devices that was designed to be open and free. These smartphone devices are being used as personal computers, i.e to access various websites, for making financial transaction etc. Along with these features, smartphones also come with certain security issues that personal computers have like malware attacks [1]. Android users can access hundreds of thousands of free or paid applications available on Android Stores. Android provide official as well as a large number of third party application stores. The popularity of Android stores also promotes malware writers to break through different these marketplaces with malicious applications. These malicious applications are mostly hidden in the vast number of benign applications that makes their detection difficult [2]. These malicious applications, such as bot, trojans and adware, are designed purposely to gain the root control of the device, or to collect sensitive information [3]. With this out of control growth of malicious contents for Android, there is a need to efficiently defend against them. But, without a complete understanding of malware behaviour and activities, it is hard to develop such a solution. This paper characterizes the current mobile malware based on their static and dynamic payloads. For the research study 100 Android malware samples in 33 different malware families which were prevalent in the year 2016 were collected. The dataset is accumulated from https://malwr.com/ and http://sanddroid.xjtu.edu.cn:8080/#overview with the help of manual or automated crawling from a variety of Android Markets. On the basis of collected malware samples, their behavioural characterization is presented. This type of classification and characterization of current Android malware is useful for their thorough understanding and to evaluate possible defence mechanisms. Next it is vital to discover whether the applications available on official Play store and other third party store are secure to download or not. This paper presents an analysis of Play store and third party Android store applications to detect malware.

The rest of the paper is as follows: Section II presents related work. Section III provides extracted permissions and other features of malware samples. Behaviour characteristics of malware families are presented section IV. Section V presents analysis of applications on Google play store and two other third party stores to identify if they are malicious or benign. Lastly, conclusion is in section VI.

II. RELATED WORK

Early techniques to classify malware like Deshotels et al. [4] used signature matching to categorize malware families. Current techniques that are enrich in syntactic information like program dependency graphs and control flow graphs, become resilient to static obfuscation methods [5], [6]. Yang et al. [5] analyzed dependency graphs build on the basis of API methods invoked by the applications. They detected malware families by encoding the feature vectors. Tangil et al. [6] proposed to apply text mining to automatically categorize malware samples. They analyzed the samples on control flow basis. Garcia et al. [7] rely on information flow analysis and sensitive API flow tracking based static analysis to detect malware. There technique was resilient against basic obfuscation methods, but it suffers from the same limitations as the other static techniques against more advanced methods. Wu et al. [8] applied machine learning algorithms on the features of the application declared in AndroidManifest.XML file and its API calls to differentiate Android malware and benign applications. Jang et al. [9] analyzed integrated system logs such system calls and their arguments to construct behavioural profiles for malware families. They used a dynamic analysis tool Droidbox for this purpose. They classified malware samples to their related families by comparing the behavioural profiles across samples. Zhou and Jiang [13] performed a timeline analysis on collected malware samples. They further characterized malware on the basis of their detailed behavior breakdown, such as the installation method, activation, and payloads. Grace et al. [14] presented potential security threats exhibited by malicious applications. They developed 'RiskRanker' an automated system that analyze dangerous behaviour posed by a particular application. Chai and Knapskog in his thesis [20] investigated the permissions demanded by Android applications, with the possibility of discovering malicious applications based merely on the information available to the user before installation of the application. During the research work, the author collected a large data set consisting of applications available on Google Play and 3 different third-party Android application stores. These applications are analyzed using manual pattern recognition and k-means clustering, focusing on the permissions they request.

III. MALWARE FEATURE EXTRACTION

Two different approaches are present to analyze a malware sample: static analysis and dynamic analysis. This section presents static and dynamic feature extraction phases to understand malware behaviour.

A. Static Feature Extraction - In static analysis, the features are extracted from the application without running that application. Certain static attributes of a malware sample such as permissions, intents, API calls etc can help the detection of similar malware. Therefore, this phase identify top permissions requested by malware database to understand their resource requirements. For example, if an application needs to use the GPS resource of device, then it must hold the ACCESS_COARSE_LOCATION or the ACCESS_FINE_LOCATION permission.

The security architecture of Android uses a permission-based security model [10]. In this model each application is associated with a group of permissions that permits the access of certain resources. Android applications files are bundled in files with apk extension that enclose all the essential classes and resources required by the applications [11]. Every application runs in its own sandbox, with a unique identifier (UID). As a result, application resources are protected from other applications and they communicate securely. Permissions cover a large set of operations, including controlling the sleep state, accessing device hardware, accessing PII, and many system operations. Applications require permissions from the users in access restricted API. According to [12], the following categories of permissions exist:

- **(i) Normal –** Granted automatically, Normal permissions don't present any risk for applications or system. Even the user is not informed when the applications are installed.
- **(ii) Dangerous –** These permissions, if used by malicious authors, may produce negative effects. If the permission request is not granted by the user, then the application is not installed.
- **(iii) Signature -** Signature permissions are useful for controlling component access to confidential applications only.
- **(iv) Signature or System –** These permissions are required for system applications and are granted only if the application requested is signed by the developer of the application.

In this permission based filtering phase static analysis on the collected malware samples is performed. Static analysis tools like ApkInspecter and Androgaurd were used for this purpose. These tools perform reverse engineering of the applications and display information like Permissions requested by applications, intents, package name, classes etc. The capability of an Android application is rigorously controlled by the permissions users grant to it. The frequently asked permissions obtained after this phase are termed as Risky permissions and permission combinations.

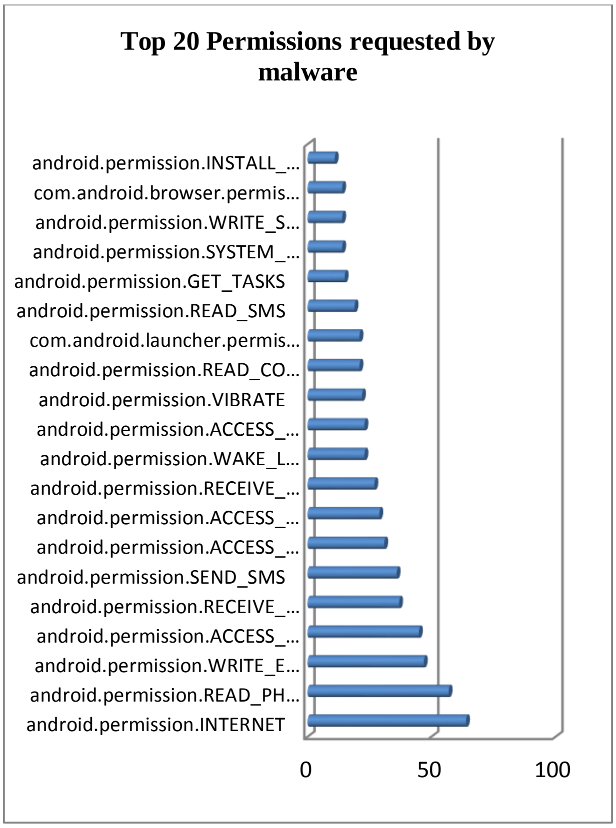
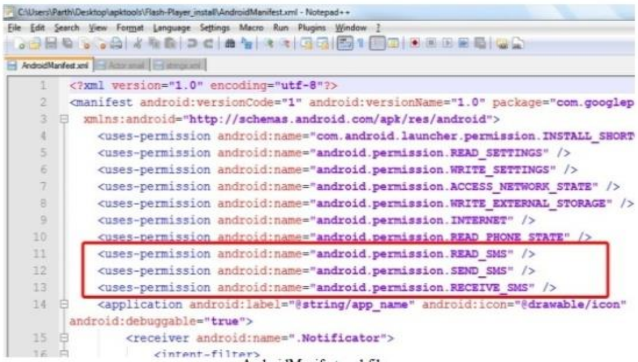


Fig 1: Top 20 permissions asked by mobile malware

Based on permission based filtering, INTERNET, READ_PHONE_STATE, WRITE_EXTERNAL_STORAGE and ACCESS_NETWORK_STATE, are widely requested permissions by malware samples. The first two are usually required to permit for the embedded ad libraries to work properly. Malicious applications are more likely to request for the SMS-related permissions, such as READ_SMS, WRITE_SMS, RECEIVE_SMS, and SEND_SMS. For instance, there are 37% samples in the dataset request for the SEND_SMS permission and 27% request for RECEIVE_SMS. RECEIVE_BOOT_COMPLETED is also frequently requested permission. 37% of the malware samples use this permission. This permission is needed by the malware to run background services without user's awareness.



AndroidManifest.xml file

Fig 2: AndroidManifest.xml File

B. Dynamic Feature Extraction - Static analysis is based on the inspection of source code without running it. With the large number of malware samples, it becomes difficult to analyze them using static analysis. Dynamic analysis or behavioural detection, on the other hand executes the sample in a controlled and remote environment to analyze its nature. It is mostly done with an automated process. This phase construct the behaviour profile of malware sample by executing it on dynamic analysis tools like TraceDroid, ScanDroid and NVISO ApkScan. Then, by comparing the behaviour profiles across samples, malware are classified into related malware families. After analyzing the reports generated by dynamic analysis tools the various features extracted are as follows:

- **(i) Information Leakage -** Information Leakage includes applications collecting IMEI, IMSI, operating system version etc. of the device and sending it to remote servers.
- **(ii) Network Activities -** network activities includes connection to command and control servers, opening of sockets and HTTP attacks etc.
- **(iii) Dynamic loading of code -** if an application is loading the code at runtime then it is not detected by static analysis tools. Hence dynamic loading of code is a suspicious behaviour and it is found during dynamic analysis that most of the malware samples are loading code at runtime.
- **(iv) File activities -** File activities include reading or updating contact list of device. Writing or deleting data on the internal and external device storage also comes under file activities.
- **(v) Suspicious API -** API's convey important behaviour about application behaviour. The frequently requested API by analyzed malware in the dataset are considered as suspicious.

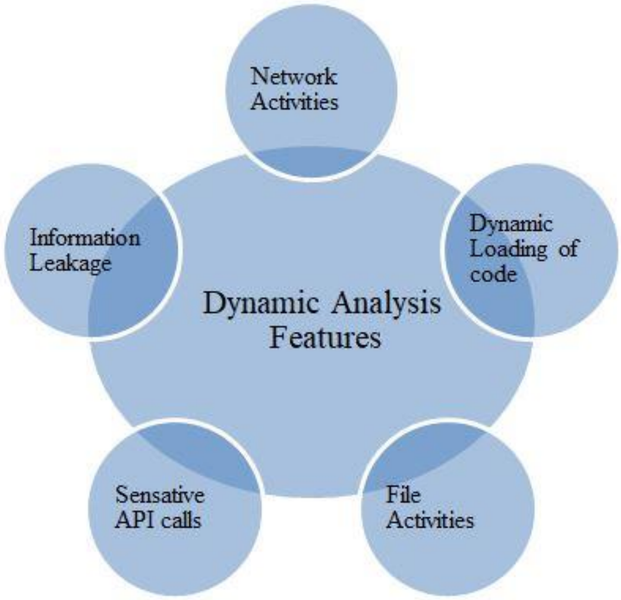


Fig 3: Features analyzed in dynamic analysis

Based on the above mentioned features/payloads the behaviour patterns of the analyzed malware samples revealed that (1) 36% of the malware families turn the compromised phones by connecting to C&C servers controlled by SMS communication. (2) Among the 33 malware families, 16 of them (48%) access built-in device features to use and send background messages or making phone calls without user awareness. (3) More than 70% malware families access personal information of the device.

IV. BEHAVIOUR CHARACTERISTICS OF MALWARE FAMILIES

The malware dataset consist of the following four types of malware.

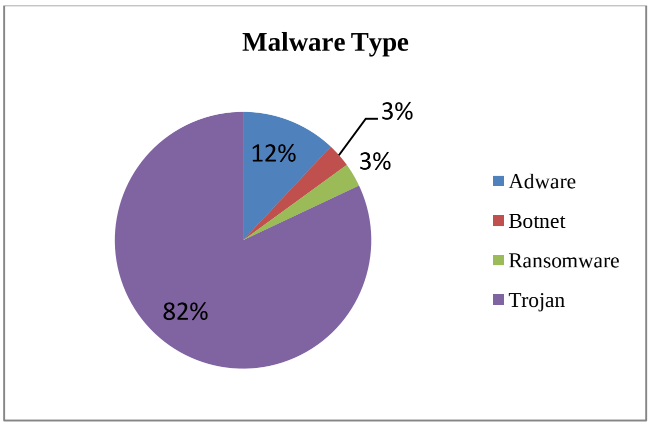


Fig 4: Type of Malware

Malware family is a set of different malware having various signature and same functionality. The malware families present in the dataset can also be partitioned by their payload features into five categories: privilege escalation, remote control, financial charges, information Leakage and repackaged.

			Information Leakage							Privilege Escalation			Remote Control	Financial		Fake App
S.No	Malware Family	Type	1	2	3	4	5	6	7	8	9	10	11	2	12	13
1	AdApperhand	Trojan	x			x					x				x	x
2	Adrd	Trojan Spy	x	x								x	x		x	x
3	Anddown	Trojan	x			x			x							
4	App2Card	Trojan sms										x				
5	Bankiller	Trojan	x	x	x				x						x	x
6	Counter Clank	Trojan	x			x		x		x					x	
7	Droid Kung Fu	Bot	x	x	x	x			x		x				x	
8	Fake app	Adware				x			x				x		**x**	
9	Fake Doc	Trojan	x	x		x		x				x	x		x	
10	Fake Inst	Trojan sms	x	x	x	x	x	x	x			x			x	x
11	Fake Run	Trojan	x	x	x	x		x	x	x		x			x	
12	Frogonal	Trojan	x				x		x				x		x	
13	Ginger Master	Trojan Spy	x						x	x	x	x	x		x	x
14	Gold dream	Trojan spy	x	x		x		x	x	x		x	x	x	x	x
15	I22hk	Trojan (Backdoor)	x						x		x	x	x			
16	Iconosys	Trojan spy	x	x											x	
17	Imlog	Trojan	x												x	
18	Jifake	Trojan sms		x								x	x	x	x	
19	Mobile TX	Trojan		x										x	x	
20	Nyleaker	Trojan Spy	x			x		x				x	x		x	
21	Opfake	Trojan SMS		x						x		x	x	x		x
22	Plangton	Trojan	x			x		x				x			x	
23	Raden	Sms trojan												x		
24	SendPay	Trojan	x											x	x	
25	Simple locker	Ransomware	x	x	x				x						x	x
26	Sms Blocker	Trojan		x							x				x	
27	Sms Bomber	Trojan		x											x	
28	SMSreg	Adware													x	x
29	Sms Spy	Trojan Sms	x	x		x			x				x		x	x
30	Sndapps	Adware Trojan	x					x					x			
31	Spy Bubble	Trojan SMS	x			x		x								
32	Spytrack	Adware													x	
33	Vdloader	Trojan	x	x		x		x		x		x			x	x
No. Of families			23	16	5	14	2	10	12	6	4	13	12	6	26	9
Percentage (%)			70	48	15	42	6	30	36	18	12	39	36	18	79	27

Table1: Behaviour characteristics of malware families

Column numbers in the above table represents the following:

1. Access device ID.
2. Send or receive sms.
3. Access camera.
4. Access geographical location.
5. Records audio.
6. Gets the MCC+MNC of the current registered operator.
7. Access SD card contents
8. Encrypt or Decrypt data.
9. Exploit Root
10. Access packages installed on the device.
11. Connects to C&C Server
12. Connects to internet
13. Repackaged: bundle with legitimate apps.

The first group (privilege escalation) covers those applications that depend on the root privilege to perform malicious activities. The repackaged group covers those fake applications that impersonate as the legitimate applications but clandestinely carry out malicious activities, for instance sending sms messages or accessing user's credentials. The third group includes applications that deliberately contain functionality that cause financial loss to users. In addition to the above payloads, most of the malware are aggressively stealing data from the infected device that include SMS messages, device identifiers as well as user accounts. Specifically, there are 16 malware families in the dataset that gather SMS messages, 23 families gather Device Id and 14 families access geographical location. Further, we are surprised to note that 12 families (36%) control the device remotely by turning it into bots. Particularly, there are 10 samples employ the HTTP-based web traffic and receive bot commands from command and control servers.

V. ANDROID APPLICATION ANALYSIS

The prime reason for Android popularity is the availability of zillions of applications available on official Google Play and third-party stores. This section identify Android application security by analyzing 1300 most downloaded applications from Google Play store and 646 from two other third party stores. Applications from Google Play store are collected with the help of a crawler. To detect malware in the collected applications permission based filtering followed by heuristic filtering is done.

A. Permission Based Filtering - In this phase with the help of ApkInspector and Androgaurd tools the collected apk were unpacked to access AndroidManifest.XML file that contains permissions requested by the application. The permissions extracted were analysed and cross verified for high occurrence across malware samples available in the template dataset. The applications detected with risky permissions were tagged as Riskware.

AppStore	Total Apps	Riskware
Play Store	1300	996
Third Party Store	646	524

Table 2: Riskware Applications

This phase may result in a high false positive ratio. The detected Riskware applications may or may not be malware. Hence the next phase plays important contribution for malware detection.

B. Heuristic Based Filtering - The Riskware applications are further analyzed in this phase for their detailed behaviour at runtime. The features of malware samples extracted in phase 3.2 acted as template here to detect similar malware functionalities. Dynamic analysis tools like NVISO ApkScan and TraceDroid were used to generate analysis reports that contain package dependency graph, network activities, disk activities, cryptographic activities etc. Manual analysis was also performed to analyze malware behaviour at runtime. Results of this phase are as follows:

AppStore	Total Apps	Riskware	Malware
Play Store	1300	996	168
Third Party Store	646	524	182

Table 3: Malware applications

Unfortunately, malware is present in third party stores and even in official Google Play store applications. Around 12.93% collected applications from Google Play store contain malware. Malware in the third party application store is more prevalent. 28. 17 % of third party store applications contain malware.

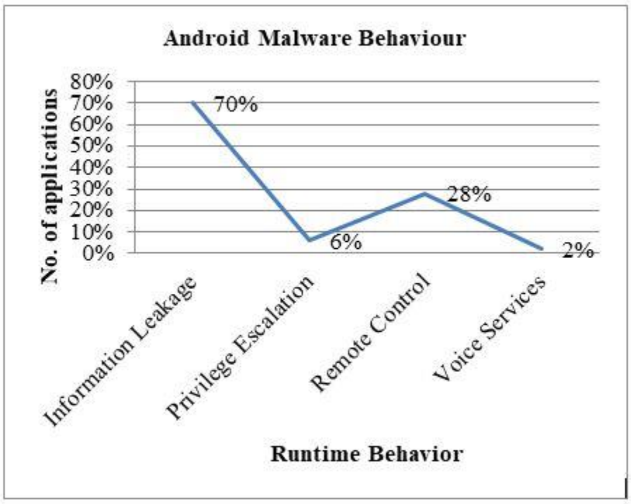


Fig 5: Android malware behaviour at runtime

Further, runtime behavioural analyses of malware reveal that 70% of applications access device information such as IMEI, phone no, contact list, location and SMS. 6% of the examined malware application gain the root exploit and breach Android security model. Many data flows directed towards code that communicates with a command and Control servers(C&C). 28% malware applications connect to a remote server. Only 2% of the total malicious applications request for CALL_PHONE permission, but none of them is misusing it.

VI. CONCLUSION

The Android platform has witnessed a huge malware growth, which is putting the sensitive information of the device at risk. Understanding the malicious actions performed by malware and to identify commonly shared behaviours by malware apps is essential for security analysis. In this paper, we collected current malware samples to monitor their runtime behaviour and identify requested permissions. The results revealed that most of the malware families send personal information of the device to the remote servers. INTERNET, READ_PHONE_STATE, SEND_SMS, RECEIVE_SMS, WRITE_EXTERNAL_STORAGE and ACCESS_NETWORK_STATE are widely requested permissions by malicious applications. Privilege escalation, remote control, financial charges, and personal information stealing are the common vulnerabilities caused by malware. These threats always come attached with legitimate applications and are hence repackaged. Further, based on static and dynamic features extracted from malware we detected similar malware in applications collected from official and third-party Android stores. The results present a worrisome scenario where no application store is safe. 70% of the detected malware applications steal personal information of the user, Hence there is a need to develop advance security solutions for Android.

VII. REFERENCES

[1]. R. Ramachandran, T., Oh, and W. Stackpole,' Android Anti-Virus Analysis', *Annual symposium on information assurance & secure knowledge management, 2012, ALBANY, New* York, pp. 12-35

[2]. M.Grace, Y. Zhou, Q. Zhang, S. Zou, and X. Jiang. 'Riskranker: scalable and accurate zero-day android malware detection', *Proceedings of International Conference on Mobile Systems, Applications, and Services*, London, UK, pp. 281–294, 2012

[3]. S. W. Hsiao, Y. S. Sun and M. C Chen,' Behavior Grouping of Android Malware Family', IEEE ICC *Proceedings of Communication and Information Systems Security Symposium*, Kuala Lumpur, Malaysia 2016

[4]. L. Deshotels, V. Notani, A. Lakhotia, "DroidLegacy: Automated familial classification of Android malware", *ACM SIGPLAN Proceedings of Program Protection and Reverse Engineering Workshop*, San Diego, CA, USA, 2014.

[5]. W. Yang, Y. Zhang, J. Li, J. Shu, B. Li, W. Hu, D. Gu, "Appspear: Bytecode decrypting and dex reassembling for packed android malware" *Proceedings of Research in Attacks Intrusions and Defenses*, *Springer*, pp. 359-381, 2015.

[6]. G. Tangil, J. E. Tapiador, P. Peris-Lopez, J. Blasco, "Dendroid: A text mining approach to analyzing and classifying code structures in android malware families", *Expert Systems with Applications*, vol. 41, no. 1, pp. 1104-1117, 2014.

[7]. J. Garcia, M. Hammad, B. Pedrood, A. Bagheri-Khaligh, S. Malek, "Obfuscation-resilient efficient and accurate detection and family identification of android malware", *Tech. Rep.*, Department of Computer Science, George Mason University, 2015.

[8]. D.-J. Wu et al., "Droidmat: Android Malware Detection Through Manifest and API Calls Tracing," *Proceedings of* Asia Joint Conference on Information Security (Asia JCIS), 2012, pp. 62–69.

[9]. Jae-wook Jang , Jaesung Yun , Aziz Mohaisen , Jiyoung Woo and Huy Kang Kim,' Detecting and classifying method based on similarity matching of Android malware behavior with profile', SpringerPlus (2016) 5:273, pp.3-23

[10]. D. Barrera, H. G. Kayacik, P. C. van Oorschot, and A. Somayaji., "A methodology for empirical analysis of permission-based security models and its application to android", *Proceedings of ACM Conference on Computer and Communications Security (CCS 2010)*, Chicago, Illinois, USA, October 4-8, 2010, pp. 73–84.

[11]. E. Burnette, "Hello, Android: Introducing Google's Mobile Development Platform", 3rd Edition. The Pragmatic Bookshelf, North Carolina, 2010, pp. 33.

[12]. P. Pocatilu, "Android Applications Security", Informatica Economica, Vol: 15 (3), pp. 163-171, 2011

[13]. Y. Zhou and X. Jiang, 'Dissecting android malware: Characterization and evolution', *Proceedings of IEEE Symposium on Security and Privacy*, San Francisco, CA, USA, pp. 95–109, 2012

[14]. M.Grace, Y. Zhou, Q. Zhang, S. Zou and X. Jiang , 'Riskranker: scalable and accurate zero-day android malware detection', *Proceedings of International Conference on Mobile Systems, Applications, and Services*, London, UK, 2012, pp. 281–294.

[15]. P.Chia and S. Knapskog., 'Information Security on the Web and App Platforms: An Economic and Socio-Behavioral Perspective,' Ph.D Thesis, Norweign university of science and technology, 2012.

Author Profiles

Sangeeta Rani is a Research Scholar at the I.K Gujral Punjab Technical University, Jalandhar, Punjab. Her research interests include social network analysis, mobile phone security and network security. She is currently working as an Assistant Professor at the Mata Gujri College, Fatehgarh Sahib, Punjab, India.

Kanwalvir Singh Dhindsa is currently working as a Professor (CSE) and Incharge, ERP at the Baba Banda Singh Bahadur Engg., College, Fatehgarh Sahib (Punjab), India, with an overall experience of 17 years in the academia and research fields of computer science and IT, He is an avid researcher having guided dissertations of many MTech and PG students and he is currently guiding seven PhD scholars. He is also on the reviewer panel of many esteemed refereed and peer-reviewed journals. His current research interests include cloud computing, web engineering, big data, internet of things, database and security, and mobile computing. He is also life member-CSI, Fellow IETE, member IEI, and life member-ISCA.