A Semi-automated Approach for Identification of Trends in Android Ransomware Literature

Tanya Gera[1], Jaiteg Singh[1(✉)], Deepak Thakur[1], and Parvez Faruki[2]

[1] Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, India
{tanya.gera,deepak.thakur}@chitkara.edu.in,
jaitegkhaira@gmail.com
[2] AV Parekh Technical Institute, Rajkot, India
parvezfaruki.kg@gmail.com

Abstract. Android ransomware is seen in the highlights of cyber security world reports. Ransomware is considered to be the most popular as well as threatening mobile malware. These are specsial malware used to extort money in return of access and data without user's consent. The exponential growth in mobile transactions from 9.47 crore in 2013–14 to 72 crores in 2016–17 could be a potential motivation for numerous ransomware attacks in the recent past. Attackers are consistently working on producing advanced methods to deceit the victim and generate revenue. Therefore, study of Android stealth malware, its detection and analysis gained a substantial interest among researchers, thereby producing sufficiently large body of literature in a very short period. Manual reviews do provide insight but they are prone to be biased, time consuming and pose a great challenge on number of articles that needs investigation. This study uses Latent Semantic Analysis (LSA), an information modelling technique to deduce core research areas, research trends and widely investigated areas within corpus. This work takes a large corpus of 487 research articles (published during 2009–2019) as input and produce three core research areas and thirty emerging research trends in field of stealth malwares as primary goal. LSA, a semi-automated approach is helpful in achieving a significant innovation over traditional methods of literature review and had shown great performance in many other research fields like medical, supply chain management, open street map etc. The secondary aim of this study is to investigate popular latent topics by mapping core research trends with core research areas. This study also provides prospective future directions for heading researchers.

Keywords: Stealth malware · Ransomware · Latent semantic analysis · Research trends · Topic solution

1 Introduction

In parallel with increasing features and benefits of Smartphone, we cannot neglect the fact that malicious software is also raising at an alarming rate. Android being an open

É. Renault et al. (Eds.): MLN 2020, LNCS 12629, pp. 265–283, 2021.
https://doi.org/10.1007/978-3-030-70866-5_18

source platform; is also susceptible to malware infections. Android users are connected to the internet for almost all the time to leverage the features of Android applications. Taking advantage of this fact, malware authors compromise the users' security by developing malicious applications. The malicious applications can harm the user by stealing their sensitive information like contacts, reading personal messages, call recording, send messages to premium rate numbers, financial loss, gain access of gallery and access to user's location etc., Popularity among users as well as consistency in malware attacks has made Android security as an interesting area in research community. Consistent on-going expansion in research dimensions of android financial malware has brought about humongous content in short span. It is generally challenging to deduce overall insight of trends in cited literature. Manual Reviews or semi-automated review processes [1–3] are two methods to make out decisions about core research areas and trends. Manual reviews can be biased at times and further it does not guarantee the inter and intra document comparison [4]. Moreover, it poses a challenge on field specific expertise to provide accurate inferences. In contrast to past works, this investigation is certifiably not a conventional synthesis on taxonomy of financial malwares or survey of existing [5–8]. This paper expects to recognize latent research trends from sufficiently large corpus of android financial malware. This work uses systematic approach called latent semantic analysis for extracting three core research areas and thirty research trends from corpus of 487 research articles. LSA is popular and proficient in enquiring important related structure.

Rest of the paper is organized as follows: second section presents related work; the third section describes the materials and research methodology adopted for collecting the research literature. The fourth section presents stepwise implementation procedure. The fifth section discusses the experimental results. The sixth section presents discussion over results obtained. The last section concludes the study with future scope for other researchers.

2 Related Work

In 2020, authors [9] performed decision making to analyze trends in blockchain technology using Word2vec-based Latent Semantic Analysis (W2V-LSA). The experimentals results confimed its usefulness and better topic modelling than tradition bibliometric methods. This approach used only 231 abstracts of blockchain related literature instead of full text documents. Systematic information retrieval using automated and semi-automated approaches in any field of research has itself become a trend. Initially, general trend analysis was perfomed first time for one-dimensional data of time [10]. However, recent advancements in text mining, information retrieval and topic modelling techniques has attracted researcher's attention towards trend analysis [11]. Text mining has also leverage machine learning algothims to enhance its capability to mine latent information [12] from user review on websites, newspaper articles, social media information analysis etc. Use of LDA is also seen in literature; but it is difficult to interpret its results without feeding a few parameters and prior knowledge of topics [13]. Other techniques like probabilistic latent semantic analysis approaches uses uni-gram format conversion of a word, which generally get fail to capture the specific context in the document [14]. In the other hand, n-gram format leads to decrease in efficiency of model due to wide dimensionality [15].

3 Material and Methods

3.1 Search Strategy and Data Gathering

With an aim to perform a systematic and detailed literature study, quality articles published in reputed journals like IEEE Transactions on Mobile Computing, IEEE Transactions on Information Forensics and Security, Computers & Security, Future Generation Computer Systems, Digital Investigation and ACM Computing Surveys have been considered. All the tasks involved are organised into research process flow. The overall outline is depicted in the Fig. 1 and discussed here below:

a.) *Article Search Strategy*

First, articles were searched, reviewed and refine on the basis of article search strategy. The keyword set used were like "malware" OR "stealth" OR "advanced" OR "persistent" OR "threats" OR "ransomware" OR "security" OR "privacy" OR "monitoring" OR "application" OR "kernel" OR "android" OR "hybrid" OR "static" OR "dynamic". Further, A fine-tuned inclusion and exclusion criteria used is explained here below in Table 1. It is an iterative process to refine the collection.

Table 1. Inclusion and exclusion criteria

Inclusion criteria	Exclusion criteria
• Articles published between 2009–2018 • Focus on Android stealth malware analysis, detection and prevention	• Articles focus on general malware • Articles focus on other operating systems like iOS, windows, Symbian, Blackberry etc.

- *Initial Paper inferences*

Under initial paper inferences, all the collected papers were thoroughly reviewed and analysed. This process majorly focusses frequency of the articles and detection approach used to augment and refine the collection. Figure 2 shows the resulting statistics as below:

- *Conduct Detailed Article inferences*

Under this task, a comprehensive manual study of research article was performed. After performing the exhaustive and extensive study of articles; a few affecting parameters came out in frame. Further, a comparison among prominent studies of Android stealth malware detection and analysis frameworks to identify objective or technique used for the study (i.e. either analysis or detection) and approach used for detection (i.e. static, dynamic or hybrid).

- *Consolidated Manual Inferences*

After compiling results of both initial and detailed level process, a few manual inferences were also made and lately will be compared with the experimental results produced by this study.

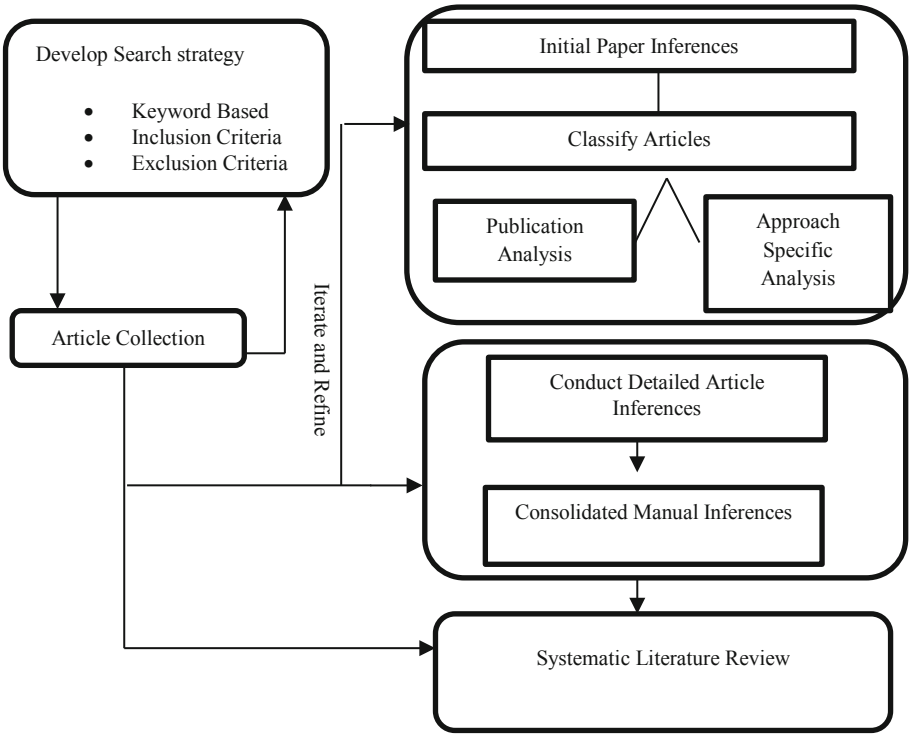

Fig. 1. Research tasks and process flow

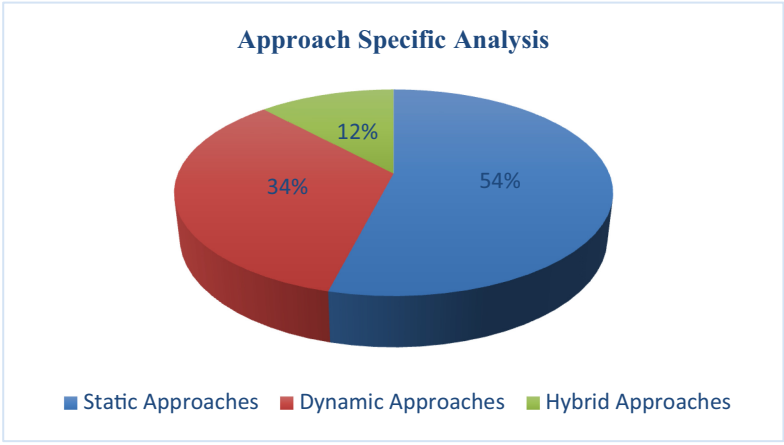

Fig. 2. Approach specific analysis

3.2 Proposed Method

The section presents the overall process flow involved in performing systematic literature study and the expected outcomes at every step are presented in Fig. 3. The whole process of systematic literature study is divided into three steps: first step is data gathering and second is to implement latent semantic analysis to extract latent topics from large gather data repository and last step is to visualize the results to present core research areas and research trends. The results of LSA will definitely help in identifying widely investigated sub-areas of financial malware attack studies.

Step 1: Data Gathering
A large corpus of research articles were gathered as discussed in detail under Sect. 3.1.
Step 2: Implementing LSA
Latent Semantic Analysis (LSA), utilizes a well-established algorithm to convert unstructured raw textual data into organized information objects, and further analyze these objects to recognize patterns for learning [3]. It employs a systematic and comprehensive approach to uncover the research trends in huge literature dataset [2]. Multiple contexts of terms belonging to different documents were collected followed by deriving the associations between related concepts that represent a latent class. A latent class can be defined as a topic solution representing multiple entities retaining similar semantics and associated patterns. The analytical tasks, which were performed in context of LSA are segregated into three modules namely Pre-LSA module, Core LSA module and Post LSA module.

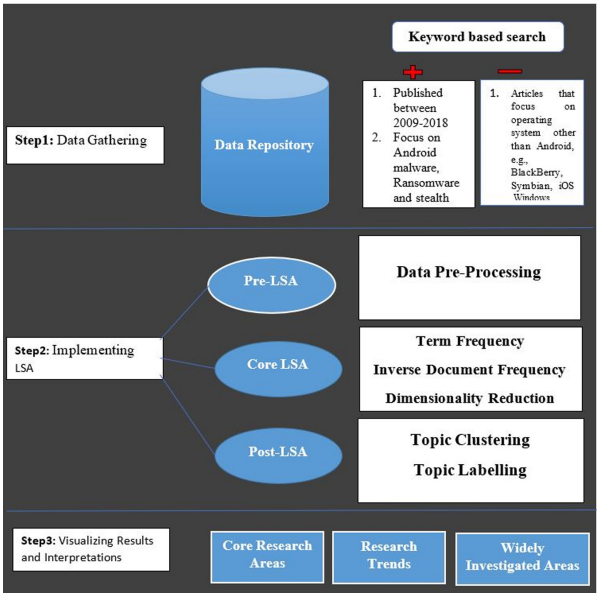

Fig. 3. Systematic literature study: process flow

(i) *Pre-LSA:* This step involves data pre-processing and generating document-term matrix of the corpus.
- Data Pre-processing: It involves tokenization, removing stop words, normalization, stemming & lemmatizing and character filtering.
- Document-Term Matrix: LSA uses bag of word (BoW) model, which results in a document-term matrix (containing term frequency in a document). LSA model typically replace raw counts in the document-term matrix with a Term Frequency-Inverse Document Frequency (TF-IDF) score. TF-IDF assigns a weight for term t in document d.

(ii) *Core-LSA:* As LSA learns latent topics by performing matrix decomposition on the document-term matrix, therefore Singular Value Decomposition (SVD) technique is used. The dimensionality reduction over matrix is implemented through an application of truncated SVD.

(iii) *Post-LSA:* Finally, terms and documents, represented in the space defined by the SVD dimensions, are analysed by implementing a specific analytic method such as queries (doscument or term comparisons), clustering, or factor analysis.

Step 3: Visualizing Results
This work aims to achieve strong mapping between dominant core research areas and research trends among the corpus of 487 abstracts of research articles related to Android financial malware attack and their studies. Latent Semantic Analysis (LSA) had been applied, as discussed in Sect. 4. As a result of which three core research areas followed by ten, twenty and thirty topic solutions were presented as research trends as described in Sect. 5.

4 Implementation

For a better understanding of overall process flow and all the related tasks while implementing LSA has been explained through a sample example in this section. Here, five sample statements have been taken as documents as in Table 2. The complete process from pre-processing to resulting scores is explained through sample documents so as to have a clear vision of implementation. However, actual results containing 487 documents are presented in Sect. 5. The implementation of LSA involves the following key steps:

***Step 1*: Pre-processing of Documents**
Pre-processing of Documents includes tokenization, removing stop words, normalization, stemming and lemmatizing, N-character filtering as explained in Table 3.

***Step 2*: Core LSA**
As the task is to mine the relevant terms that provide useful or quality information about the document, the document-term matrix has to be replaced by TF-IDF weights for further processing of the matrix. TF-IDF weight is a measure to interpret the importance of a term to a document in a collection of the large corpus. TF-IDF works on the fact that relevant words are not necessarily frequent words. The TF-IDF weight is build-up of two terms that need to be calculated beforehand:

- **TF (Term Frequency)**: It provides the Term Frequency (TF) with normalized value, which can be computed as the division of the number of occurrences of a term in a document and the total number of terms in that document, refer Eq. 1. Sample TF scores in context with an example taken are shown in Table 4.

$$TF(t,d) = \frac{Number\ of\ occurences\ of\ term\ t\ appears\ in\ document\ d}{Total\ number\ of\ terms\ in\ the\ document} \tag{1}$$

- **IDF (Inverse Document Frequency)**: IDF takes care of the importance of each term within the document. IDF is calculated as a log of the division of count of documents in the corpus divided by the count of documents where the specific term appears. However, it is quite obvious that some terms, like "is", "the", "of", and "that" or certain domain-specific words, will appear many a time but may not have much importance. Thus, arises a need to weigh down the importance of most occurred terms while scaling up the rare ones. Therefore, IDF scores are important and can be computed as given in Eq. 2. IDF scores for sample documents are presented in Table 5.

$$IDF(t,d) = log\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ \ term\ t\ in\ it} \tag{2}$$

Hence, after the calculation of TF and IDF scores, the final document-term matrix with TF-IDF scores is calculated with the following Eq. 3.

$$w_{t,d} = TF_{t,d} \times Log\frac{N}{df_t} \tag{3}$$

Table 2. Sample statements

Documents	Sample statements
Document 1	Crypto ransomware encrypts the personal files/folders
Document 2	Lock screen ransomware locks the screen and demands payment
Document 3	Attackers use tactics like spam emails to spread the ransomware
Document 4	Ransomware authors blackmail the victims of the ransomware attack
Document 5	Ransomware can easily evade from signature-based mechanisms

(where t denotes the terms; d denotes each document; N denotes the total number of documents).

Consider Table 6 above which represents the document-term matrix with TF-IDF scores for previously stated example, a term will have more TF-IDF value when its occurrences across the *document* are more but less across the *corpus*. Let's take an example of a domain-specific word i.e. "malware" which will be fairly a very common word in the whole corpus but may appear often in a document hence, it will not have a high TF-IDF score. However, the word "permissions" may appear frequently in a document, and appears less in the rest of the corpus, it will have a higher TF-IDF score.

The corpus for the TF-IDF calculation will be the collection of all pre-processed documents. TF-IDF's implementation was used that is included in the sklearn's package feature_extraction.text of python library and can be easily called using the class TfidfVectorizer.

As shown in the example given above, a large corpus of the Android security dataset resulted in high dimensional TF-IDF matrix. High dimensional matrix is generally expected to be redundant and noisy. Therefore, to uncover the relationship among the words and documents and capture the latent topics within the corpus, dimensionality reduction was performed, as detailed in the following section.

Table 3. Sample outcomes after Pre-processing

S. no	Pre-processing steps	After pre-processing
a.)	**Tokenization** • Converts Large Chunks of text to sentences • Sentences to Words	['Malware', 'application', 'reads', 'the', 'unique', 'device', 'identifier', 'to', 'track', 'the', 'user', 's', 'device']
b.)	**Removing Stop Words** • Remove stop words • Remove Common words	['Malware', 'application', 'reads', 'unique', 'device', 'identifier', 'track', 'user', 'device']
c.)	**Normalization** • Standard Formatting • Upper case to lower case • Numbers to word equivalents	Malware application reads the unique device identifier to track the user s device
d.)	**Stemming and Lemmatizing** • Reduces total number of unique words • Converts the words to their word stem • Past and Future tenses to present • Third form to first form	['malwar', 'applic', 'read', 'uniqu', 'devic', 'identifi', 'track', 'user', 'devic']
e.)	**N-Character Filtering** • Words less than the length 4 were omitted	['malwar', 'applic', 'read', 'uniqu', 'devic', 'identifi', 'track', 'user', 'devic']

Table 4. TF scores

Documents	TF scores
Doc 1	{'crypto': 0.083, 'ransomwar': 0.167, 'encrypt': 0.083, 'person': 0.083, 'file': 0.083, 'folder': 0.083, 'applic': 0.083, 'similar': 0.083, 'function': 0.083, 'call': 0.083, 'packag': 0.083}
Doc 2	{'lock': 0.286, 'screen': 0.286, 'ransomwar': 0.143, 'demand': 0.143, 'payment': 0.143}
Doc 3	{'attack': 0.143, 'tactic': 0.143, 'like': 0.143, 'spam': 0.143, 'email': 0.143, 'spread': 0.143, 'ransomwar': 0.143}
Doc 4	{'ransomwar': 0.333, 'author': 0.167, 'blackmail': 0.167, 'victim': 0.167, 'attack': 0.167}
Doc 5	{'ransomwar': 0.167, 'easili': 0.167, 'evad': 0.167, 'signatur': 0.167, 'base': 0.167, 'mechan': 0.167}

Table 5. IDF scores

Terms	IDF score	Terms	IDF score	Terms	IDF score
Applic	2.098612	evad	2.098612	ransomwar	1.000000
Attack	1.693147	file	2.098612	Screen	2.098612
Author	2.098612	folder	2.098612	signatur	2.098612
Base	2.098612	function	2.098612	Similar	2.098612
blackmail	2.098612	like	2.098612	Spam	2.098612
call	2.098612	lock	2.098612	Spread	2.098612
crypto	2.098612	mechan	2.098612	Tactic	2.098612
demand	2.098612	packag	2.098612	Victim	2.098612
easili	2.098612	payment	2.098612		
email	2.098612	person	2.098612		
encrypt	2.098612				

Step 3: **Dimensionality Reduction Using Singular Vector Decomposition**
TF-IDF matrices produce in the previous step clearly has high dimensional data which is difficult to interpret. So, to convert it into low dimensional vector space, singular vector decomposition is applied. This step converts the tf-idf matrix into two matrices term loading matrices and document loading matrices as shown in Fig. 4. Term loading matrix represents association between terms and topics whereas, document loading matrix represents association between documents and topics. The corresponding scores for both matrices are shown in Table 7 and 8 respectively. Each topic in the matrices signals towards a research theme in the whole corpus. It is totally in hands of the researcher to

Table 6. TF-IDF scores

Terms	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5
Applic	0.302777	0.000000	0.000000	0.000000	0.000000
Attack	0.000000	0.000000	0.332773	0.377851	0.000000
Author	0.000000	0.000000	0.000000	0.468337	0.000000
Base	0.000000	0.000000	0.000000	0.000000	0.437393
blackmail	0.000000	0.000000	0.000000	0.468337	0.000000
Call	0.302777	0.000000	0.000000	0.000000	0.000000
Crypto	0.302777	0.000000	0.000000	0.000000	0.000000
Demand	0.000000	0.312698	0.000000	0.000000	0.000000
Easily	0.000000	0.000000	0.000000	0.000000	0.437393
Email	0.000000	0.000000	0.412464	0.000000	0.000000
Encrypt	0.302777	0.000000	0.000000	0.000000	0.000000
Evad	0.000000	0.000000	0.000000	0.000000	0.437393
File	0.302777	0.000000	0.000000	0.000000	0.000000
Folder	0.302777	0.000000	0.000000	0.000000	0.000000
function	0.302777	0.000000	0.000000	0.000000	0.000000
Like	0.000000	0.000000	0.412464	0.000000	0.000000
Lock	0.000000	0.625395	0.000000	0.000000	0.000000
mechan	0.000000	0.000000	0.000000	0.000000	0.437393
Package	0.302777	0.000000	0.000000	0.000000	0.000000
payment	0.000000	0.312698	0.000000	0.000000	0.000000
Person	0.302777	0.000000	0.000000	0.000000	0.000000
ransomwar	0.288550	0.149002	0.196541	0.446330	0.208420
Screen	0.000000	0.625395	0.000000	0.000000	0.000000
signatur	0.000000	0.000000	0.000000	0.000000	0.437393
Similar	0.302777	0.000000	0.000000	0.000000	0.000000
Spam	0.000000	0.000000	0.412464	0.000000	0.000000
Spread	0.000000	0.000000	0.412464	0.000000	0.000000
Tactic	0.000000	0.000000	0.412464	0.000000	0.000000
Victim	0.000000	0.000000	0.000000	0.468337	0.000000

alter the value of number of topic solutions they want at a particular time. Researchers also face a significant issue at times during implementing this step i.e. multiple documents can be mapped into same topic solutions. So, to avoid an iterative process of k-mean clustering is further applied to visualise the results clearer and easy to interpret.

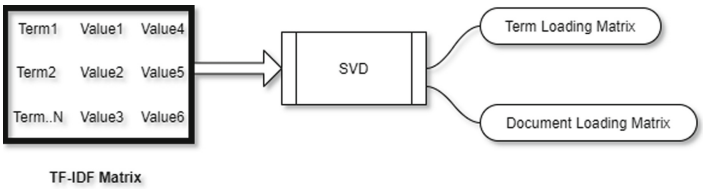

Fig. 4. Applying SVD over TF-IDF matrix

Table 7. Term loading matrix scores

Terms	Topic 0	Topic 1	Topic 2	Topic 3	Topic 4
Applic	0.107961	0.068837	0.068550	0.258226	−0.076493
Attack	0.353090	−0.247212	−0.110429	−0.088438	0.094649
Author	0.251933	−0.095740	−0.035791	−0.016441	0.401476
Base	0.124356	0.158389	0.316281	−0.221523	−0.062988
blackmail	0.251933	−0.095740	−0.035791	−0.016441	0.401476
Call	0.107961	0.068837	0.068550	0.258226	−0.076493
Crypto	0.107961	0.068837	0.068550	0.258226	−0.076493
demand	0.067287	0.226574	−0.195981	−0.057421	−0.028939
Easily	0.124356	0.158389	0.316281	−0.221523	−0.062988
Email	0.185713	−0.210672	−0.101084	−0.093175	−0.284161
Encrypt	0.107961	0.068837	0.068550	0.258226	−0.076493
Evad	0.124356	0.158389	0.316281	−0.221523	−0.062988
File	0.107961	0.068837	0.068550	0.258226	−0.076493
Folder	0.107961	0.068837	0.068550	0.258226	−0.076493
function	0.107961	0.068837	0.068550	0.258226	−0.076493
Like	0.185713	−0.210672	−0.101084	−0.093175	−0.284161
Lock	0.134575	0.453147	−0.391961	−0.114842	−0.057878
mechan	0.124356	0.158389	0.316281	−0.221523	−0.062988
Package	0.107961	0.068837	0.068550	0.258226	−0.076493
payment	0.067287	0.226574	−0.195981	−0.057421	−0.028939

(*continued*)

Table 7. (*continued*)

Terms	Topic 0	Topic 1	Topic 2	Topic 3	Topic 4
Person	0.107961	0.068837	0.068550	0.258226	−0.076493
ransomwar	0.522796	0.057412	0.040377	0.053107	0.130504
Screen	0.134575	0.453147	−0.391961	−0.114842	−0.057878
signatur	0.124356	0.158389	0.316281	−0.221523	−0.062988
Similar	0.107961	0.068837	0.068550	0.258226	−0.076493
Spam	0.185713	−0.210672	−0.101084	−0.093175	−0.284161
Spread	0.185713	−0.210672	−0.101084	−0.093175	−0.284161
Tactic	0.185713	−0.210672	−0.101084	−0.093175	−0.284161

Table 8. Document loading matrix scores

	Topic 0	Topic 1	Topic 2	Topic 3	Topic 4
Doc 1	0.477735	0.224989	0.219206	0.797172	−0.193947
Doc 2	0.288304	0.717045	−0.606811	−0.171642	−0.071047
Doc 3	0.603249	−0.505456	−0.237279	−0.211149	−0.528886
Doc 4	0.720724	−0.202300	−0.073990	−0.032813	0.658088
Doc 5	0.380924	0.358358	0.700109	−0.473394	−0.110552

Step 4: **Topic Clustering and Topic Labelling**
Researchers can vary the number of topic solutions, but cannot go beyond number of documents at maximum. Here, a problem may arise in which multiple documents can be mapped into same topic solutions. So, to avoid an iterative process of k-mean clustering is further applied to visualise the results clearer and easy to interpret. K-Means clustering is used to form clusters of semantically related terms. Here, in this study five, ten, twenty and thirty topic solutions are successfully discovered and presented in subsequent section. Loading weights of the terms for every topic solution are sorted in descending order so as to give suitable labels for high loading values with help of subject field experts. This is being done for all five, ten, twenty and thirty topic solutions. Though we tried to provide results for fifty topic solution but it resulted in overlapping and repetitions of terms. However, this could also be due the dataset chosen.

5 Experimental Results

This study identifies three core research areas and research themes trends across the wide literature set of 487 articles focussing on Android stealth malware. Semantic analysis results in three dominant core research areas. However, iterative process of varying the number of topic solutions results in ten, twenty and thirty topic solutions. Each topic

solution is considered as $\mathbf{TS}_{x.y}$ where x denotes solution number whereas y is yth factor of the xth number topic solution. For example, TS10.2 should be considered as 2nd factor of ten solutions. The Detained description along with their publication counts as discussed in subsequent section.

The graphs plotted for all topic solutions also provides information about the publication count for each topic label during three different time periods within 2009–2019, Figs. 4 and 5. The publication count associated with each topic solution represents the importance of the corresponding research area within that topic solution. Further, to uncover the research trends and future directions in the field of Android security, thirty topic solutions were also discovered as shown in Table 12. The semantic mapping between fifty topic solutions and five core research areas helps to identify research trends within each core research area, as presented in Table 10.

5.1 Core Research Areas

Systematic literature Analysis over Android stealth malware corpus results in identifying three core areas as Table 9.

Table 9. Core research areas

Topic no.	Topic label	Top loading terms
TS3.1	App Structure Monitoring	signature, bytecode, graph, context, dalvik, flow, permission, component, control, library, program, service, method, object, entry, event, field, code, data, path
TS3.2	App Behaviour Monitoring	kernel, privilege, escalation, policy, control, enforcement, security, exploit, memory, vulnerability, library, native, context, component, linux, mechanism, access, resource, sandbox, virtual
TS3.3	Hybrid Level Monitoring	dynamic, analysis, static, cloud, taint, application, instruction, execution, component, sensitive, bytecode, android, library, program, native, object, string, dalvik, class, event

Core Research Areas (TS3.1) is found to be the most trending area, whereas (TS3.2) behavioural analysis of applications seems to be utilised and became in trend from 2014 onwards. TS3.3 produces promising results in analysing and detecting of smart malwares. Likewise, Table 10 and 11 shows ten topic solution and twenty topic solutions, their sensible labels and their count values as follows:

Table 10. Ten topic solutions

Topic no	Topic label	2009–2013	2014–2019	2009–2019
TS10.1	Emulator Based Analysis	23	26	49
TS10.2	Dynamic Code Loading	19	32	51
TS10.3	High Battery Consumption	22	21	43
TS10.4	Context Monitoring	21	30	51
TS10.5	API Call Monitoring	18	31	49
TS10.6	Dalvik Byte Code Analysis	21	17	38
TS10.7	Permission Based Analysis	22	31	53
TS10.8	Classification Based on App Behavior	19	38	57
TS10.9	Graph Based Analysis	13	29	42
TS10.10	Feature Based Analysis	23	31	54

Table 11. Twenty topic solutions

Topic no	Topic label	2009–2013	2014–2019	2009–2019
TS20.1	Obfuscated Code Analysis	5	6	11
TS20.2	Privacy Leakage Monitoring	19	7	26
TS20.3	Hybrid Analysis	5	24	29
TS20.4	Pattern Assessment	9	12	21
TS20.5	Permission Based Analysis	22	20	42
TS20.6	Kernel Level Check	12	18	30
TS20.7	Signature Based Analysis	16	19	35
TS20.8	Classification Based on App Behavior	11	21	32
TS20.9	Dynamic Code Loading	8	21	29
TS20.10	Emulator Based Analysis	7	14	21
TS20.11	Taint Analysis	4	12	16
TS20.12	Graph Based Analysis	12	13	25
TS20.13	Flow Monitoring	3	9	12
TS20.14	API Call Monitoring	6	12	18
TS20.15	User Interactions	3	13	16
TS20.16	Context Monitoring	12	22	34
TS20.17	Feature Based Analysis	4	7	11
TS20.18	Dalvik Byte Code Analysis	8	15	23
TS20.19	High Battery Consumption	4	19	23
TS20.20	Text Based Analysis	4	29	33

The results show that under ten topic solutions (TS10.8) Classification Based on App Behavior is most widely used method for analysis and detection of Android malware. Specifically, TS10.8 became in trend between the year 2014–2018. However, (TS10.7) i.e. Permission based analysis remained trending during 2009–2014. Under twenty topic solutions, results showed in Table 11 clearly depicts (TS20.5), (TS20.6), (TS20.7), (TS20.8) and (TS20.16) became most trending factors that pose a significantly impact on malware detection process.

5.2 Research Themes/Trends

After an extensive iterative process, thirty topic solutions considered to be the major core research trends/themes across wide corpus of Android ransomware. Table 12 shows list of topic solutions along with their count of articles falling under the same solution.

Table 12. Research trends

Topic no	Label	2009–2013	2014–2019	2009–2019
TS30.1	Kernel Level Check	6	15	21
TS30.2	Dynamic Code Loading	4	19	23
TS30.3	Classification Based on App Behavior	9	8	17
TS30.4	Obfuscated Code Analysis	2	7	9
TS30.5	Ransomware detection	5	10	15
TS30.6	User Interaction	15	11	26
TS30.7	Runtime Analysis	3	6	9
TS30.8	Emulator Based Analysis	4	4	8
TS30.9	Privacy Leakage Monitoring	2	7	9
TS30.10	Flow Monitoring	4	8	12
TS30.11	Sandboxing Techniques	1	15	16
TS30.12	Text-based analysis	2	7	9
TS30.13	Fuzz Testing	8	4	12
TS30.14	Pattern Assessment	6	13	19
TS30.15	Graph Based Analysis	3	13	16

(*continued*)

Table 12. (*continued*)

Topic no	Label	2009–2013	2014–2019	2009–2019
TS30.16	Function Call Monitoring	3	6	9
TS30.17	Hybrid Analysis	4	12	16
TS30.18	Permission Based Analysis	10	11	21
TS30.19	Dalvik Byte Code Analysis	5	24	29
TS30.20	High Battery Consumption	5	18	23
TS30.21	File Operation	1	12	13
TS30.22	Context Monitoring	7	12	19
TS30.23	API Call Monitoring	7	22	29
TS30.24	VM Monitoring	2	10	12
TS30.25	Taint Analysis	6	15	21
TS30.26	Signature based Similarity	2	10	12
TS30.27	Context Monitoring	2	14	16
TS30.28	Privilege Escalation	9	11	20
TS30.29	Feature Based Analysis	6	9	15
TS30.30	System Events	1	10	11

Among thirty core research area, research trends like Dynamic Code Loading (TS30.2), Dalvik Byte Code Analysis (TS30.19), API Call Monitoring (TS30.23). Results also shows that these topic solutions are found to be most widely investigated areas. The areas remained in trend throughout the year 2014–2019.

5.3 Widely Investigated Areas

Another aim of this study was to identify topmost widely investigated sub-areas inside each core research areas. The results of experimentation helped to interpret significant analysis patterns. Though static analysis is more popular than dynamic but it may be due to corpus selected for analysis. That dominance of behaviour-based study is more than structure-based studies. Here, permission based, graph based, context based and signature based turned out to be most impactful sub-areas under App structural monitoring (TS3.1). As depicted in Fig. 5; out of 145 articles, 6% of them focussed on permission-based analysis, likewise 9% were of signatures-based similarity. Likewise, other widely investigated sub-areas can be observed from Fig. 5 below:

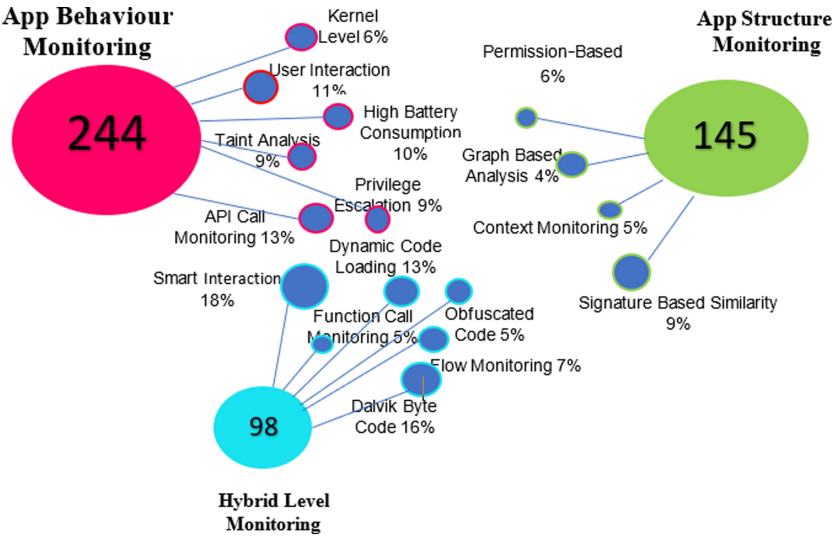

Fig. 5. Widely investigated sub-areas

6 Discussion

To prove the significance of this study, results framed out after complete LSA experimentation is compared with a few inferences made manually from literature review:

- Need of robust hybrid solutions:
  Manual reviews states, there exists comparatively less hybrid approaches for malware detection when compared to Static and Dynamic approaches. Our study also shows hybrid solutions are found to be only 21%, comparatively very less than app structure monitoring and app behaviour monitoring.
- Stealthy Behaviour:
  Recent reported ransomware silently installed its packages in mobile phones. They first gained administrative rights to exploit the user's privacy and subsequently encrypts device data. To detect such malware kernel level scanning is required (Ahmadian et al. 2015; Tam et al. 2017; Rashidi and Fung 2015). Our results state out of 244 articles, kernel level check was 9%.
- Intense hardware requirement for real time analysis:
  Analysing infections on the mobile phone itself requires high consumption of CPU power, battery and memory, while analysis on the server might suffer from high communication overhead and delayed response (AI-rimy et al. 2018; Alazab et al. 2012; Xu et al. 2016). These clearly sign towards the requirement of designing a hybrid framework for effective detection and predictions.

7 Conclusion

Manual literature review may give bias and incomplete interpretations. This study aims to review a large literature set of 487 articles systematically to identify latent research themes. Implementing LSA helped to three core research areas and thirty research trends. Results suggested that overall analysis and detection process of ransomware revolves around three core areas i.e. app structure, app behaviour and hybrid monitoring. Among thirty core research area, research trends like dynamic code loading, dalvik byte code analysis, API call monitoring. Results also shows that these topic solutions are found to be most widely investigated areas. The areas remained in trend throughout the year 2014–2019 and are expected to be in remain in future also. Our study also shows hybrid solutions are found to be only 21%, comparatively very less than app structure monitoring and app behaviour monitoring. For future researchers, analysis at kernel level is need of the hour to detect financial attacks. Researchers can apply same methodology in one or more other research fields by little or almost no changes. LSA can help researchers to identify research trends in other fields also.

References

1. Lee, S., Song, J., Kim, Y.: An empirical comparison of four text mining methods. J. Comput. Inf. Syst. **51**(1), 1–10 (2010)
2. Evangelopoulos, N., Zhang, X., Prybutok, V.R.: Latent semantic analysis: five methodological recommendations. Eur. J. Inf. Syst. **21**(1), 70–86 (2012)
3. Delen, D., Crossland, M.D.: Seeding the survey and analysis of research literature with text mining. Expert Syst. Appl. **34**(3), 1707–1720 (2008)
4. Yalcinkaya, M., Singh, V.: Patterns and trends in building information modeling (BIM) research: a latent semantic analysis. Autom. Constr. **59**, 68–80 (2015)
5. Becher, M., Freiling, F.C., Hoffmann, J., Holz, T., Uellenbeck, S., Wolf, C.: Mobile security catching up? Revealing the nuts and bolts of the security of mobile devices. In: 2011 IEEE Symposium on Security and Privacy, pp. 96–111 (2011)
6. Enck, W.: Defending users against smartphone apps: techniques and future directions. In: Jajodia, S., Mazumdar, C. (eds.) ICISS 2011. LNCS, vol. 7093, pp. 49–70. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25560-1_3
7. Suarez-Tangil, G., Tapiador, J.E., Peris-Lopez, P., Ribagorda, A.: Evolution, detection and analysis of malware for smart devices. IEEE Commun. Surv. Tutorials **16**(2), 961–987 (2013)
8. Faruki, P., Bharmal, A., Laxmi, V., Gaur, M.S., Conti, M., Rajarajan, M.: Evaluation of android anti-malware techniques against dalvik bytecode obfuscation. In: 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, pp. 414–421 (2014)
9. Kim, S., Park, H., Lee, J.: Word2vec-based latent semantic analysis (W2V-LSA) for topic modeling: a study on blockchain technology trend analysis. Expert Syst. Appl. **152**, 113401 (2020)
10. Kivikunnas, S.: Overview of process trend analysis methods and applications. In: ERUDIT Workshop on Applications in Pulp and Paper Industry, pp. 395–408 (1998)
11. Kang, H.J., Kim, C., Kang, K.: Analysis of the trends in biochemical research using latent dirichlet allocation (LDA). Processes **7**(6), 379 (2019)
12. Kim, Y.M., Delen, D.: Medical informatics research trend analysis: a text mining approach. Health Inform. J. **24**(4), 432–452 (2018)

13. Alghamdi, R., Alfalqi, K.: A survey of topic modeling in text mining. Int. J. Adv. Comput. Sci. Appl. (IJACSA) **6**(1), 147–153 (2015)
14. Lu, Y., Zhai, C.: Opinion integration through semi-supervised topic modeling. In: Proceedings of the 17th İnternational Conference on World Wide Web, pp. 121–130 (2008)
15. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. J. Mach. Learn. Res. **3**, 1137–1155 (2003)