A-SIT PLUS GmbH

Detecting Inconsistencies between Android App Descriptions and Permissions

Detecting Inconsistencies between Android App Descriptions and Permissions

Autor:
Emina Ahmetovic
emina.ahmetovic@iaik.tugraz.at
Datum: 03.02.2023

Abstract/Zusammenfassung:
Android users are offered a vast number of apps that provide a variety of functionalities and assistance in everyday life. While the functionality of the applications can have a strong impact on the privacy of the user, permissions are introduced as a mechanism that protects users' assets by asking for explicit consent when accessing privacy-sensitive data. Nevertheless, users often struggle to find a connection between requested permissions and the description of the app.

To reliably identify if the need for permission is justified is a challenging task that we aim to tackle in this project. We propose a novel machine-learning approach that predicts app behavior based on the information provided by developers. We create a dataset with 46 000+ app descriptions and permissions. Furthermore, we design a model using a state-of-the-art Transformer that identifies whether the need for permission is outlined in the description of the app and to what extent.

Inhaltsverzeichnis

Contents

1. Introduction

With 3.55 million Android apps - a number that is expected to grow, Google Play is the leading app market that has with over 2.8 billion active users and a global market share of 75 percent[1]. Without a doubt, apps are prevalent in everyday life. However, the success story makes Android apps a common target for malicious applications, that profit from users' private data. The popularity of mobile devices in everyday life often leads to privacy incidents on mobile devices via permission models. While the apps tend to offer more functionalities, they also have more means to invade users' privacy. Very often, users have the challenging task of denying or allowing permission requests from their app. Privacy implications that apps carry have been in the spotlight, mostly since the data breach incidents caused by the apps. As previous studies warn, granting certain permission requests can lead to privacy leaks that users are not aware of (Felt A. P., 2011) (Felt A. P., 2012) (Li, 2021) (Almuhimedi, 2015). While a large portion of users understand the privacy implications that come with permissions carry, they struggle to correctly assess if that permission is necessary for that app. In many cases, users cannot find an explanation for the use of permissions that can heavily affect their data (Liu B. a., 2016), (Shen, 2021).

In this project, we aim to assess to what extent the claimed permissions are actually introduced to users. Permissions are one of the main security mechanisms on a mobile OS that play an important role in protecting users' data. When users grant certain permissions, apps get access to the private data of the users. For this reason, we argue that apps should explain access to private resources and that users should understand dangerous permissions well. Thus, we investigate the app descriptions to predict the declared permissions. While this study can also be done using another type of metadata, we focus on the app description as the informal, user-friendly channel that can assist users in better understanding the app's privacy and security functionalities. We argue that descriptions are the appropriate mean that should introduce the use of dangerous permissions to users.

1.1 Contribution

In this project, we perform an application description analysis to find inconsistencies between description and permission in android apps. Our study is different from the previous one because we, to the best of our knowledge, are the first ones to use a state-of-the-art Transformer neural model to perform the permission-to-description fidelity. To summarize our contribution:

- We have implemented a crawler to create a dataset of 46 000+ application descriptions and permissions. Our search was based on the applications from TOP popularity that belong to the 49 categories.
- We design and implement the deep neural network that performs the multiclassification problem and predicts the permission based on the textual document. For this task, we used the Transformer architecture with the pre-trained model of Bert.
- Finally, we evaluate our model on real-world apps against the seven permission groups that are assigned to the dangerous level.

The rest of this study is structured as follows: In section 2, we will explain the mechanism of permissions and what kind of permissions we will tackle in this report. In section 3, will give a quick overview of the background of the Transformer deep neural network model. In section 4, we explain how we created a dataset. In section 5, we outline the results of our evaluation. In section 6, we discuss the limitations and future work. In section 7, we give an overview of the related work.

[1] https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

2. Background

Android applications are distributed centrally; this means that developers, in addition to the apps, also publish metadata, such as descriptions, reviews, and similar. These metadata have been used together with NPL and machine learning techniques in the application metadata analysis. The novel analysis, in contrast to the other type of analysis, such as static or dynamic, has shown success in tacking the description-to-permission fidelity, which is a term in literature for detecting how well apps described behavior matched the actual behavior. Many studies have performed such analysis. The idea behind the study is related to the descriptions as an important communication channel between developers and end users.

In the following sections, we will introduce the important background techniques relevant to our study.

2.1 Permissions on Android

The permission model is introduced as a mechanism that gives users control over their data and therefore offers protection of their privacy. An app needs to obtain explicit permission from a user to access or use a certain resource. Based on the scope of the restricted actions and access to the restricted resources, Android defines different permission types: dangerous or runtime permissions and normal or install-time permissions. Install time permissions[2] execute actions with a minimal impact on the other app or system; thus, they are automatically granted by the system at the install time (a time when a user installs the app). The install time permissions are not displayed to the users by default but rather can be found in the Play Store details. The normal and signature permission belong to the install-time permissions.

On the other hand, runtime permissions[3] such as SMS, CONTACTS, CALENDAR, LOCATION, and similar are assigned to the dangerous protection level since they allow apps to access the restricted data and perform the restricted operation. They are requested at runtime (right before performing the restricted action) and need explicit approval from the user. In contrast to the install time permissions, users can revoke runtime permissions at any time.

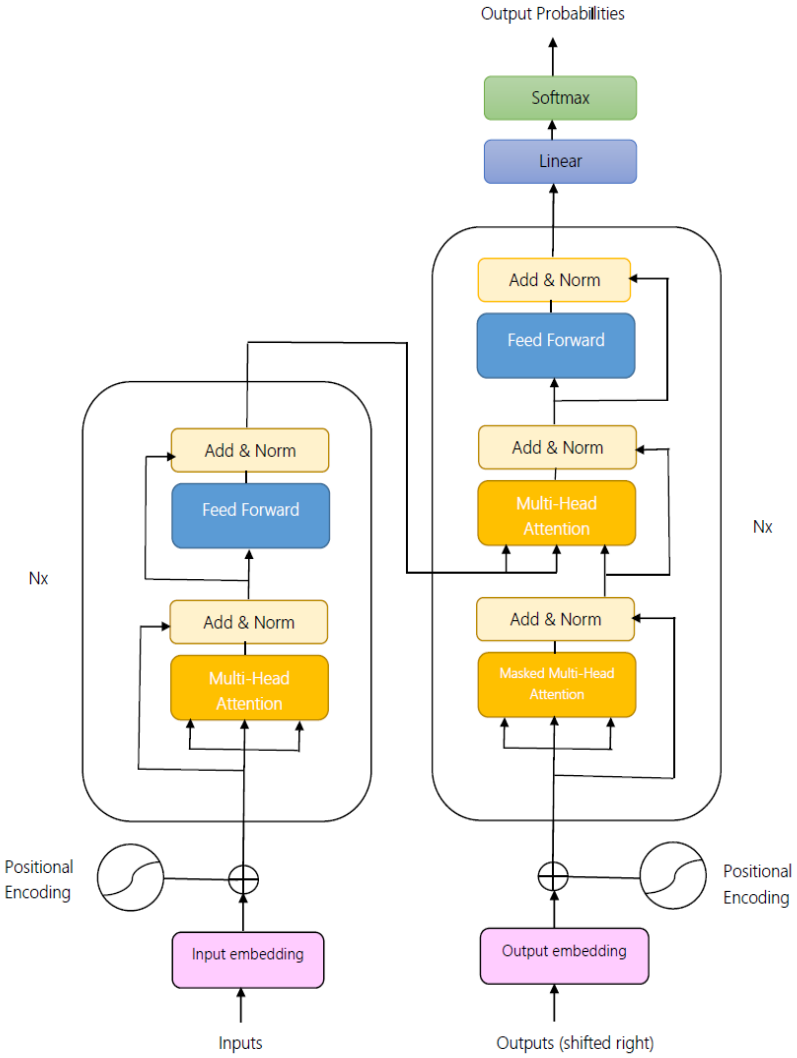
3. Attention and Transformer model

Recurrent neural networks (Graves, 2012), such as long short-term memory and gated recurrent neural networks, for many years, were state-of-the-art in sequence modeling tasks such as language modeling and machine translation. Nevertheless, they are well known for their inherited limitations in terms of sequence processing, which is one of the reasons why they are being replaced with the Transformer. The Transformer (Vaswani, 2017) is a neural network model that adopts the self-attention mechanism which in contrast to RNNs and CNNs defines global attention between input and output. This means that with self-attention each word in the sentence attends to every other word in the sentence. Same as recurrent neural networks, transformers also process input data sequentially, but in contrast to RNNs, transformers process the entire input data at once, which is highly parallelizable in matrix form and reduces the training time. The Transformer is based on encoder-decoder architecture, as shown in picture 1.

The encoder consists of two layers: multi-head self-attention and a fully-connected feed-forward network. In the beginning, it takes the positional encoding that contains the order of the sequence and input embedding as the input information. At every layer, input encoding is generated from the previous encoder and fed to the self-attention mechanism that weights their relevance and generates the output encoding. The output encoding is passed to the feed-forward neural network that computes the output encoding individually. While the encoder generates encoding and passes it to the next encoder, the decoder receives the encoding and generates the

[2] https://developer.android.com/guide/topics/permissions/overview#install-time
[3] https://developer.android.com/guide/topics/permissions/overview#runtime

output sequence. The decoder consists of three layers: multi-head self-attention, fully-connected feed-forward network, and encoder-decoder self-attention. The architecture of the decoder is similar to the encoder; however, it has an additional layer that performs multi-head attention over the encodings generated by the encoders.



Picture 1. The encoder-decoder architecture of the Transformer.

3.1 Attention Mechanism

The traditional seq-to-seq model that consists of an encoder and decoder based on the GRU or LSTM units, generally speaking, aimed to transfer the input sequence arbitrary length into the output sequence also arbitrary length (Cho, 2014). However, the disadvantage of this approach was the inability to remember long sentences. For example, in text generation, if the distance between output and input is too long, the input part will be forgotten. The attention mechanism emerged as an efficient mechanism to resolve the problem of long distances between sources and targets. Attention, roughly speaking, means paying attention to the different words in a sentence. In contrast to the gated RNNs, where a state vector contains a representation of the data prior to the current token, the attention layer can access all previous states and weigh them according to relevance. Thanks to the shortcuts (Bahdanau, 2014), (Luong, 2015) created between the context vector and the entire source input, their weights can be calculated for each output element individually. While an attention mechanism was already used together with RNN architecture (cite), the Transformer, replacing the recurrent layers, relies solely on the attention mechanism and accomplishes state-of-the-art results in many NLP tasks, such as text translation.

4. Dataset

To create a dataset of the description, we have implemented a crawler that obtains the most popular application metadata. Besides descriptions and permissions necessary for our study, we have also crawled the additional information such as application ID, review, data safety section, ranking, category and similar, privacy policy, and similar. A JSON object below shows one sample of the fetched app metadata.


```
{
        "genreID": "SHOPPING",
        "score": "0.0",
        "ratings": "0",
        "permissions": [
                "Location"
        ],
        "appID": "com.fressnapf.mobileapp",
        "description": " 'Maxizoo/Fressnapf is the partner for you and your pet - right in your pocket.  The most important features in the app: - Save 5% on every purchase in the store and online with the Friends discount*. - Save easily with exclusive coupons in our stores - Shop online and find and order your favorite products - Create an individual profile for your pet - Exciting content and articles about your pet - Find your favorite store near you and have all the offers from your store at your fingertips  Maxizoo/Fressnapf has been the contact for all pet-related questions since 1990. Customers and their pets trust us because we also love pets and are committed to their well-being. In our stores, you&#39;ll find everything you need for pets, whether it&#39;s pet food or pet accessories. With the exclusive coupons, you can also save money easily and conveniently. In our online store, you can access our extensive product range at any time from anywhere. Whether in the store or online: your pet is always our priority.  *See terms and conditions per country: - Germany: https://www.fressnapf.de/friends/ - Austria: https://www.fressnapf.at/friends/ - France: https://www.maxizoo.fr/friends/ - Poland: https://www.maxizoo.pl/friends/  For suggestions for improvement, or if you have problems with our app, you can contact us at app-android.Team@fressnapf.com.  Thank you and happy shopping! Your Maxizoo/Fressnapf App Team',\n  ",
        "title": "Maxizoo / Fressnapf",
        "privacy policy url": "https://www.fressnapf.de/app/datenschutzhinweise/"
}
```


The initial dataset contained 80 000 application metadata; however, after removing the non-English descriptions, or the ones that were too short, we used 46 431 app metadata from the overall 49 categories. The distribution of the applications over categories is shown in the table below.

SHOPPING	1310	AUTO_AND_VEHICLES	1168
ENTERTAINMENT	1062	NEWS_AND_MAGAZINES	1142
SPORTS	1138	MEDICAL	1143
VIDEO_PLAYERS	837	HOUSE_AND_HOME	1056
SOCIAL	873	COMICS	401
PHOTOGRAPHY	841	GAME_PUZZLE	886
ART_AND_DESIGN	895	GAME_BOARD	854
FINANCE	1190	GAME_EDUCATIONAL	804
COMMUNICATION	909	PARENTING	683
BOOKS_AND_REFERENCE	948	GAME_ARCADE	849
MAPS_AND_NAVIGATION	1187	GAME_CASUAL	863
HEALTH_AND_FITNESS	1150	GAME_SIMULATION	864
TOOLS	1097	LIBRARIES_AND_DEMO	443
WEATHER	1001	GAME_WORD	827
FOOD_AND_DRINK	1445	GAME_TRIVIA	856
BUSINESS	1296	GAME_RACING	828

EVENTS	611	GAME_ADVENTURE	853
MUSIC_AND_AUDIO	932	GAME_SPORTS	890
LIFESTYLE	1175	GAME_ACTION	811
TRAVEL_AND_LOCAL	1257	GAME_MUSIC	853
EDUCATION	1037		

Table 1. Application categories and the number of the app that belong to the group.

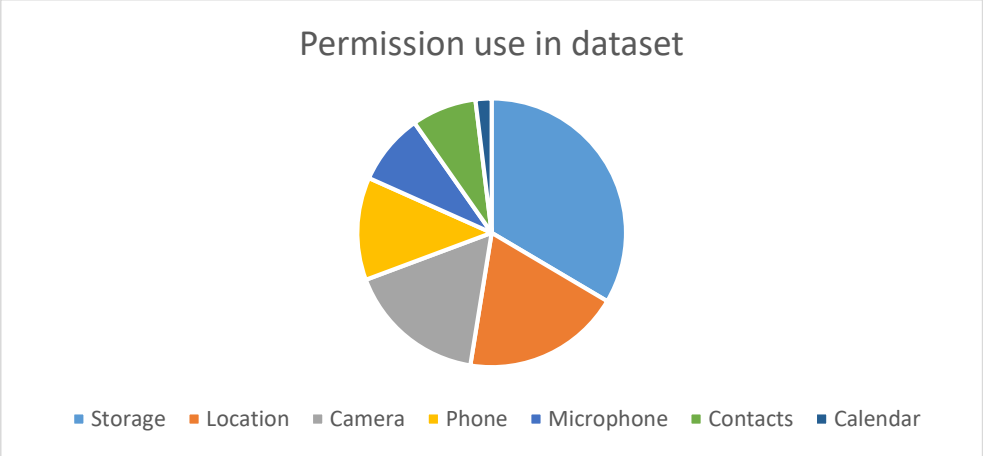
4.1 Permissions in a dataset

The aim of our study is to find inconsistencies between permissions declared by the app and the description of the application. We consider only permissions assigned to the dangerous level due to their potential impact on privacy. The convenience of taking permission groups instead of individual permission is that permissions are granted on the group level, meaning that if the app needs READ_EXTERNAL_STORAGE, the user needs to grant the entire group Storage, as that is what the app will be requesting. We evaluate our solution against seven permission groups outlined in table 2.

calendar	READ_CALENDAR WRITE_CALENDAR	1672
camera	CAMERA	14472
contacts	READ_CONTACTS WRITE_CONTACTS GET_ACCOUNTS	6729
location	ACCESS_FINE_LOCATION ACCESS_COARSE_LOCATION ACCESS_BACKGROUND_LOCATION ACCESS_MEDIA_LOCATION	16432
microphone	RECORD_AUDIO	7407
phone	READ_PHONE_NUMBERS READ_PHONE_STATE CALL_PHONE READ_CALL_LOG WRITE_CALL_LOG ADD_VOICEMAIL USE_SIP PROCESS_OUTGOING_CALLS ANSWER_PHONE_CALLS ACCEPT_HANDOVER	10668
storage	READ_EXTERNAL_STORAGE WRITE_EXTERNAL_STORAGE	28907

Table 2. Dangerous permission groups and the individual permissions that belong to the group. The third column represents the number of permission requests in the dataset.

The permission distribution over the entire dataset shows that the most frequent permission request is for the Storage, following Location, and Camera. The least common permission request is for Calendar. Picture 2 represents the permission distribution in our dataset. Initially, we have included two additional permissions, namely SMS and Sensors; however, due to their very sparse use, we have decided to exclude them from our study.



Picture 2. The permission usage in the dataset.

5. Evaluation

We structure our problem as a multi-label classification problem, where the input is an application description text, and the output is a one-hot encoded vector. We one-hot encode the vector and put ones in places where the application is requesting permission and 0 if the app does not request permission. One example of inputs and outputs:

Input text: *"Product Features Amazon Shopping offers apponly benefits to help make shopping on Amazon faster and easier than shopping on your desktop Never miss a delivery Get realtime tracking and delivery notifications so you know where your package is and when it arrives Know exactly what youre purchasing Full 360 product view lets you see items from every angle View in you room makes sure it fits by using your phones camera and VR so you can see it in your space Well notify you when items go on sale Just tap the heart icon to save items to Your Lists and well alert you of price drops so you dont miss a deal Never forget your password Save time by staying securely signed in If you prefer to sign out use facial or fingerprint identification to sign back in Connect with us when it works best for you Live chat support is open 24 hours 7 days a week Once youve started a chat it stays that way for 24 hours so you dont have to start your support session from the beginning Well find that item for you Not sure of an items brand or where to but it Just tap the scan icon in the search bar take a picture of the item or its barcode and well find it for you Product Description Browse search view product details read reviews and purchase millions of products We deliver to 100 countries in as quickly as 35 days Whether youre buying gifts reading reviews tracking orders scanning products or just shopping Amazon Shopping app offers more benefits than shopping on Amazon via your desktop Important Note Regarding Permissions Please note that the Amazon Shopping app requires access to the following services to operate properly Contacts Allows you to send Amazon gift cards to your contacts or invitation to install the Amazon app Camera Allows the Amazon app to access your camera on the device You can use your camera to find products by scanning the cover or its barcode to add gift cards and credit cards or to add pictures in the product reviews Flashlight Allows the Amazon app to turn on the flashlight You can use the flashlight to find products with the camera feature even in lowlight or dark conditions Microphone Allows the Amazon app to access your microphone to use your voice to search and interact with your Assistant Location Allows the Amazon app to access your location to help you discover local offers and select addresses fast Account Allows you to share products on Amazon with your friends and families through Facebook or other social networks Phone Allows the Amazon app to prepopulate the Amazon Customer Service number on your phones keypad Storage Allows the Amazon app to store your preferences so that some features can load and run faster on the device WiFi This permissions is used when setting up either a Dash Button or Dash Wand using the Amazon Shopping app The Amazon App for Tablets is available on Google Play Search for quotAmazon Tabletquot to install the app and begin shopping For customers located within the European Union United Kingdom Brazil or*

Turkey By using this app you agree to Amazons Conditions of Use applicable for your country Please also see the applicable Privacy Notice Cookies Notice and InterestBased Ads Notice for your country Links to these terms and notices can be found in the footer of your local Amazon homepage For all other customers By using this app you agree to the applicable Amazon Conditions of Use eg wwwamazoncomconditionsofuse and Privacy Notice eg wwwamazoncomprivacy for your country Links to these terms and notices can be found in the footer of your local Amazon homepage"

One-hot encoding: [0, 1, 1, 1, 1, 1, 1], where labels are ['Calendar', 'Camera', 'Contacts', 'Location', 'Microphone', 'Phone', 'Storage'] respectively.

Before we process data, we perform the preprocessing application description. The preprocessing consists of removing text that contains HTML tags, and other special characters like emojis. Furthermore, we split the data into training-validation-test sets, each containing 80-10-10 percent of the original data. We perform the transfer-learning approach on the BERT[4] model with the following hyperparameters:

Hyperparameter	Value	Description
num_train_epochs	10	Number of epochs to train our model, one epoch is going over the whole training dataset
gradient_accumulation_steps	2	Number of steps to accumulate gradients, we accumulate the gradients for 2 steps and apply them to our learning parameters
manual_seed	43	The manual seed for reproducibility purposes
max_seq_length	512	The maximum number of words in the description
use_early_stopping	True	To prevent overfitting, we use early stopping
early_stopping_delta	0.01	We check if our metrics on the validations data are at least 0.01 different than our metric on the training data
early_stopping_metric	mcc	Matthews Correlation Coefficient is the metric used for early stopping
early_stopping_metric_minimize	False	We want to maximize the metric
early_stopping_patience	3	When overfitting is detected, we wait for 3 steps until stopping the training
save_model_every_epoch	False	For storage optimization, we do not store model every epoch

Table 3. The table depicts the hyperparameters we used in our model.

The previous hyperparameters are specific to the simple transformers library[5], that is used for the model training. We obtain the following results on the test:

	precision	recall	F1-score	support
Calendar	0.32	0.35	0.33	135
Camera	0.72	0.76	0.74	1428
Contacts	0.47	0.55	0.51	660
Location	0.80	0.75	0.78	1640
Microphone	0.55	0.61	0.58	761
Phone	0.49	0.55	0.52	1080
Storage	0.92	0.68	0.78	2880
Micro avg	0.70	0.67	0.69	8548
Macro avg	0.61	0.61	0.61	8548

Table 4. The table shows the obtained results.

[4] https://huggingface.co/docs/transformers/model_doc/bert

[5] https://github.com/ThilinaRajapakse/simpletransformers

As our dataset is not balanced, displaying values for accuracy may not provide a good metric and would result in misleading interpretation. Instead, to correctly interpret the obtained results, we use the following metric:

- *Precision* measures how many true positives that are made are correct (true positives / (true positives + false positive))
- *Recall*, or sensitivity, measures the number of true positives over all positives in data (true positives / (true positives + false negative))
- *F1 score*, as the harmonic mean of the Precision and Recall
- *Support*, as the number of all true positives the model identified
- *Micro average*, computes the average metrics by calculating the results of all classes
- *Macro average*, computes the metrics independently for all classes and averages the results

From the results above, we can conclude that our precision ranges from 61% to 70%. We can also note that the best results we obtain for Storage and Location, which are by far the most used permissions. On the other hand, the lowest value we have for the Calendar, due to its rare use. The macro value for F1 is the lowest, which is an indication that the dataset is imbalanced to some extent.

5.1 Comparison with other models

The prior work that aim to calculate description-to-permission fidelity was done mainly using different deep learning architectures based on recurrent neural networks or convolutional neural networks. Comparing other models with ours is challenging due to many reasons; authors either do not open their dataset, the existing dataset is outdated, or the number of permissions differs. However, in table 3, we outline three prior frameworks that used different models and their corresponding results.

	Apps	Permissions	Model	Results			
AC-Net: Assessing the Consistency of Description and Permission in Android Apps (Feng, 2019)	1 415	16	RNN GRU	ROC-AUC		0.974	
				PR-AUC		0.669	
FCDP: Fidelity Calculation for Description-to-Permissions in Android Apps (Wu, 2020)	64 265	16	RNN LSTM	ROC-AUC		0.9679	
				PR-AUC		0.5992	
Understanding Privacy Awareness in Android App Descriptions Using Deep Learning (Feichtner, 2020)	77 758	9	CNN		Precision	Recall	F1 score
				Micro	81%	65%	77%
				Macro	77%	56%	70%
Our model	46 000+	7	Transformer		Precision	Recall	F1 score
				Micro	70%	67%	69%
				Macro	61%	61%	61%

Table 5. Comparison of the results from previous work.

We can see that our results are comparable with the existing ones. Transformer, in contrast to CNN or RNN, comes with additional benefits such as a variety of pre-trained models with state-of-the-art results that can be reused, and simple implementation.

6. Limitations and future work

One of the limitations of the current model is the size of the dataset. Increasing data in the dataset would improve the results, as the Transformer model performs better with a large amount of data. In addition, future work should compare different pre-trained models. As we used the BERT pre-trained model, it would be beneficial to compare the performances of other pre-trained models such as XLNet[6]. Regarding the hyperparameters, future work also includes tuning the maximum sequence length to capture more extended context, as it is observed that some descriptions could be very long.

7. Related work

One of the first research done in the domain of application description analysis was done by (Pandita, 2013). The authors have proposed a framework called Whyper, which checks if the need for dangerous permissions used by the app is introduced in the description. It is one of the first studies that involve Natural Language Processing (NLP) for extracting semantic meaning from descriptions to detecting text that refers to permissions. They have addressed the limitations of the keyword-based approach, such as confounding meaning and semantic inference, and built the semantic model by manually analyzing Android API documents. Their dataset includes 581 apps, and Whyper is evaluated against three permissions: READ CALENDAR, READ CONTACTS, and RECORD AUDIO.

The term description-to-permission fidelity is introduced by Qu et al. (Qu, 2014) where authors argue that users should gain an intuitive idea about the security and privacy functionality of the application by reading descriptions and that descriptions should give an idea about the requested permissions. They propose a fully automated framework AutoCog, where they use NLP techniques to infer permission use from app descriptions. However, their key component for semantic extraction is Explicit Semantic Analysis (ESA) which leverages an extensive knowledge base (i.e., Wikipedia), in contrast to using a dictionary-based corpus like WordNet used in Whyper.

In a study done by Gorla et al. (Gorla, 2014), the authors propose the CHABADA framework, which uses description and the called APIs to detect anomalies and check application behavior. Chabada uses the Latent Dirichlet Allocation (LDA), a tool and technique for topic modeling, on descriptions to identify the main topic for each application and to further cluster applications by related topics. In each cluster, they extract sensitive APIs and use an unsupervised clustering algorithm to find outliers with respect to API usage. Their dataset of 22500+ applications from the Google Play Store is available for reproducibility purposes.

To answer why the app descriptions fail to refer to the use of privacy-sensitive resources, Watanabe et al. (Watanabe, 2015) introduce the ACODE framework that leverages a keyword-based approach. ACODE uses a two-stage filter combining static code and keyword-based text analysis. They evaluate ACODE on a dataset with 200 000 apps that are from the official as well as from third-party markets. Authors report comparable results; however, unlike other studies, they include both English and Chinese language.

[6] https://huggingface.co/docs/transformers/model_doc/xlnet

References

Almuhimedi, H. a. (2015). Your location has been shared 5,398 times! A field study on mobile app privacy nudging. *Proceedings of the 33rd annual ACM conference on human factors in computing systems*, (pp. 787--796).

Bahdanau, D. K. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Cho, K. V. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint*.

Feichtner, J. a. (2020). Understanding privacy awareness in android app descriptions using deep learning. *Proceedings of the tenth ACM conference on data and application security and privacy*, 203--214.

Felt, A. P. (2011). Android permissions demystified. *Proceedings of the 18th ACM conference on Computer and communications security*, (pp. 627--638).

Felt, A. P. (2012). Android permissions: User attention, comprehension, and behavior. *Proceedings of the eighth symposium on usable privacy and security*, (pp. 1--14).

Feng, Y. a. (2019). AC-Net: Assessing the consistency of description and permission in Android apps. *IEEE Access (7)*, 57829--57842.

Gorla, A. a. (2014). Checking app behavior against app descriptions. *Proceedings of the 36th international conference on software engineering*, 1025--1035.

Graves, A. (2012). Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, 37--45.

Li, R. a. (2021). Android Custom Permissions Demystified: From Privilege Escalation to Design Shortcomings. *2021 IEEE Symposium on Security and Privacy (SP)* (pp. 70--86). IEEE.

Liu, B. a. (2016). Follow my recommendations: A personalized privacy assistant for mobile app permissions. *Twelfth symposium on usable privacy and security (SOUPS 2016)*, (pp. 27--41).

Liu, Y. M. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* .

Luong, M.-T. H. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* .

Pandita, R. X. (2013). {WHYPER}: Towards automating risk assessment of mobile applications. *USENIX Security Symposium (USENIX Security 13)*, 527–542.

Qu, Z. a. (2014). Autocog: Measuring the description-to-permission fidelity in android applications. *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 1354--1365.

Shen, B. a. (2021). Can Systems Explain Permissions Better? Understanding Users' Misperceptions under Smartphone Runtime Permission Model. *30th USENIX Security Symposium (USENIX Security 21)*, (pp. 751--768).

Vaswani, A. N. (2017). Attention is all you need. *Advances in neural information processing systems 30* .

Watanabe, T. a. (2015). Understanding the inconsistencies between text descriptions and the use of privacy-sensitive resources of mobile apps. *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*, 241--255.

Wu, Z. a.-J. (2020). FCDP: Fidelity calculation for description-to-permissions in Android apps. *IEEE Access (9)*, 1062--1075.