ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет безопасности информационных технологий

КУРСОВАЯ РАБОТА

По дисциплине:

«*Обеспечение информационной безопасности мобильных устройств*»

На тему:

«*Разработка системы для журналирования и проверки журнала доступа к телефонно-адресной книге*»

Выполнила:
студентка группы N33471
Ершова Ксения Александровна

_____

Проверил преподаватель:
Таранов Сергей Владимирович

_____

Отметка о выполнении:

_____

Санкт-Петербург
2021 г.

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студентка Ершова Ксения Александровна

(Фамилия И.О.)

Факультет Безопасности Информационных Технологий

Группа N33471

Направление (специальность) 10.03.01. - Технологии защиты информации

Руководитель Ищенко Алексей Петрович- преподаватель (квалификационная категория “преподаватель практики”)

(Фамилия И.О., должность)

Дисциплина Обеспечение информационной безопасности мобильных устройств

Наименование темы Разработка системы для журналирования и проверки журнала доступа к телефонно-адресной книге

Задание Обеспечение безопасности персональных данных из телефонно-адресной книги, путем проверки приложений на экспорт данных на сторонние платформы

Краткие методические указания

1. Стандарт ISO/IEC 27005:2018 "Information technology – Security techniques – Information security risk management" («Информационная технология. Методы и средства обеспечения безопасности. Менеджмент рисков информационной безопасности»).

2. Стандарт NIST SP 800-30 "Guide for Conducting Risk Assessments" («Руководство по проведению оценки рисков»)

Содержание пояснительной записки Курсовая работа содержит введение, функциональную модель протокола передачи персональных данных «ИКС», перечень актуальных уязвимостей ИБ для протокола, модификацию протокола, оценку эффективности предлагаемой модификации, заключение и список литературы

Рекомендуемая литература

1. Бельский В.С. Протокол обмена персональными данными: ИКС / В.С. Бельский, И.Ю. Герасимов, К.Д. Царегородцев, И.В. Чижов // International Journal of Open Information Technologies, 2020. – 23 с

2. Приказ Федеральной службы по надзору в сфере связи, информационных технологий и массовых коммуникаций от 05.09.2013 №996 «Об утверждении требований и методов по обезличиванию персональных данных».

Руководитель Ищенко Алексей Петрович

(Подпись, дата)

Студент Ершова Ксения Александровна

(Подпись, дата)

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

ГРАФИК ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

Студентка Ершова Ксения Александровна
(Фамилия И.О.)

Факультет Безопасности Информационных Технологий

Группа N33471

Направление (специальность) 10.03.01. - Технологии защиты информации

Руководитель Ищенко Алексей Петрович- преподаватель (квалификационная категория “преподаватель практики”)
(Фамилия И.О., должность)

Дисциплина Обеспечение информационной безопасности мобильных устройств

Наименование темы Разработка системы для журналирования и проверки журнала доступа к телефонно-адресной книге

№ п/п	Наименование этапа	Дата завершения		Оценка и подпись руководителя
		Планируемая	Фактическая	
1	Разработка и согласование ТЗ	9.06.2021		
2	Анализ актуальных уязвимостей ИБ для объекта защиты	11.06.2021		
3	Разработка модели объекта защиты	12.06.2021		
4	Разработка модификации протокола	13.06.2021		
5	Защита курсовой работы	15.06.2021		

Руководитель Ищенко Алексей Петрович
(Подпись, дата)

Студентка Ершова Ксения Александровна
(Подпись, дата)

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

АННОТАЦИЯ НА КУРСОВУЮ РАБОТУ

Студент Ершова Ксения Александровна

(Фамилия И.О.)

Факультет Безопасности Информационных Технологий

Группа N33471

Направление (специальность) 10.03.01. - Технологии защиты информации

Руководитель Ищенко Алексей Петрович- преподаватель (квалификационная категория “преподаватель практики”)

(Фамилия И.О., должность)

Дисциплина Обеспечение информационной безопасности мобильных устройств

Наименование темы Разработка системы для журналирования и проверки журнала доступа к телефонно-адресной книге

ХАРАКТЕРИСТИКА КУРСОВОГО ПРОЕКТА (РАБОТЫ)

1. Цель и задачи работы ☐ Предложены студентом ☑ Сформулированы при участии студента ☐ Определены руководителем

2. Характер работы ☐ Расчет ☐ Конструирование ☑ Моделирование ☐ Другое

3. Содержание работы

анализ актуальных уязвимостей ИБ, сравнение уязвимостей операционных систем, разработка алгоритма для проверки программ на вредоносность и в частности экспорт данных из телефонно-адресной книги

4. Выводы

операционные системы далеки от идеала, из-за чего нам нужно обращать внимание на все среды, т.к работая в сфере кибербезопасности, можно столкнуться с комплексом сложных проблем

Руководитель Ищенко Алексей Петрович

(Подпись, дата)

Студентка Ершова Ксения Александровна

(Подпись, дата)

«___»_____20___г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Передача персональных данных и предотвращение их раскрытия являются распространенными проблемами в цифровом мире. Согласно статистике, в 2020 году мобильные приложения были загружены на устройства пользователей более 218 миллиардов раз. Большую часть времени, проведенного в цифровом пространстве, было потрачено на программы в смартфонах и планшетах. Для многих людей мобильные приложения стали частью жизни: банкинг, мессенджеры, бизнес-приложения, личные кабинеты сотовых операторов- при современном ритме жизни, эти приложения используются практически ежедневно. Согласно данным nafi.ru мобильными приложениями банков в 2020 году пользовался 51% россиян, и количество пользователей возросло почти на 30% по сравнению с 2018 годом. По данным Positive Technologies, количество уникальных киберинцидентов в 2020 году выросло на 51% в сравнении с 2019 годом.

Разработчики стараются сделать приложения визуально привлекательными и максимально удобными для пользователей. Люди охотно устанавливают мобильные приложения, регистрируются в них, вводят свои данные, но мало кто из этих пользователей задумывается о безопасности доверенных данных.

Для исследования будет рассмотрена операционная система Android, которая является одной из самых распространённых систем во всем мире. Из-за своего технологического влияния, открытого исходного кода и возможности установки приложений от третьих лиц без какого-либо централизованного управления Android в последнее время стал целью вредоносного ПО. Даже если он включает механизмы безопасности, последние новости о злонамеренных действиях и уязвимостях Android указывают на важность продолжения разработки методов и структур для повышения безопасности.

Чтобы предотвратить атаки вредоносных программ, исследователи и разработчики предложили различные решения безопасности, применяя статический анализ, динамический анализ и искусственный интеллект. Обычно анализируют киберугрозы, используя два метода: статический анализ и динамический анализ.

Статический анализ: включает в себя методы, которые позволяют нам получать информацию о программном обеспечении, которое мы хотим проанализировать, не запуская его, одним из примеров является изучение кода, их вызовов, ресурсов и т. д.

Динамический анализ: это другой подход, идея которого состоит в том, чтобы анализировать киберугрозу во время ее выполнения, другими словами, получать информацию о ее поведении, некоторые ее функций — сетевые потоки.

Актуальность исследования определяется возникшим увеличением случаев кибератак и **возможностью** обеспечить защиту персональных данных путем разработки соответствующего алгоритма, основанного на поиске уязвимостей и дальнейшего усиления безопасности.

Объект исследования: операционная система Android.

Предмет исследования: алгоритм, позволяющий извлекать и создавать CSV-файл с информацией о разрешениях приложений. С помощью которого в дальнейшем можно сопоставить каждый пакет приложений Android со списком разрешений.

Цель исследования: повышение уровня защиты персональных данных в процессе их взаимодействия со сторонними приложениями.

Задачи для достижения цели исследования:

– изучение набора данных Malgenome.;

– выделение параметров и показателей, которые задействованы вредоносными ПО и обычными приложениями;

– разработка алгоритма, обучающего различные классификаторы обнаруживать вредоносное ПО;

– обучение алгоритма отличать вредоносное ПО от остальных приложений;

– анализ полученных результатов и составление отчетной документации.

Анализ актуальных уязвимостей ИБ для объекта защиты

Актуальные исследования по оценке ИБ среди мобильных приложений и операционных систем

- Уязвимости высокого уровня риска обнаружены в 38% мобильных приложений для iOS и в 43% приложений для платформ под управлением Android.
- Большинство проблем безопасности являются общими для обеих платформ. Небезопасное хранение данных — основной недостаток, он выявлен в 76% мобильных приложений. Под угрозу попадают пароли, финансовая информация, персональные данные и личная переписка.
- Хакеру редко требуется физический доступ к смартфону, чтобы украсть данные: 89% уязвимостей могут быть проэксплуатированы с использованием ВПО.
- Большинство недостатков связаны с ошибками в механизмах защиты (74% и 57% — для приложений на iOS и Android соответственно, 42% — для серверных частей). Такие уязвимости закладываются еще на этапе проектирования, а их устранение потребует внесения существенных изменений в код.
- Риски возникают не только из-за отдельно взятых уязвимостей на клиенте или сервере; зачастую угрозы обусловлены несколькими, казалось бы, незначительными недостатками в разных частях мобильного приложения, которые в совокупности могут приводить к серьезным последствиям, вплоть до финансового ущерба для пользователей и репутационных потерь для производителя.
- Успех кибератаки на мобильное приложение напрямую зависит от того, насколько внимательно сам пользователь относится к сохранности своих данных. Предпосылкой ко взлому могут стать повышенные привилегии или загруженные из неофициального источника программы.

Оценка рисков

- 60% уязвимостей сосредоточены в клиентской части
- 89% уязвимостей могут быть проэксплуатированы без физического доступа к устройству
- 56% уязвимостей могут эксплуатироваться без административных прав (jailbreak или root)

Для Android несколько чаще встречаются приложения с критически опасными уязвимостями, чем программы для iOS (43% против 38%). Однако эта разница несущественна, и общий уровень защищенности клиентских частей мобильных приложений для Android и iOS примерно одинаков. Около трети всех уязвимостей в клиентских частях мобильных приложений для обеих платформ имеют высокий

уровень риска. На рисунке 1 показана максимальная степень риска уязвимостей (указана доля клиентский частей).

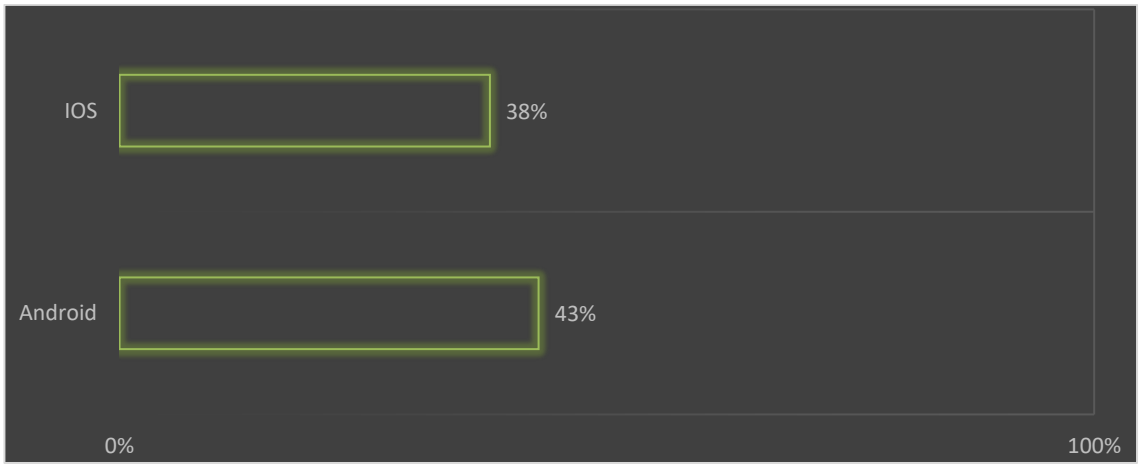


Рисунок 1

На рисунке 2 показана доля уязвимостей различной степени риска.

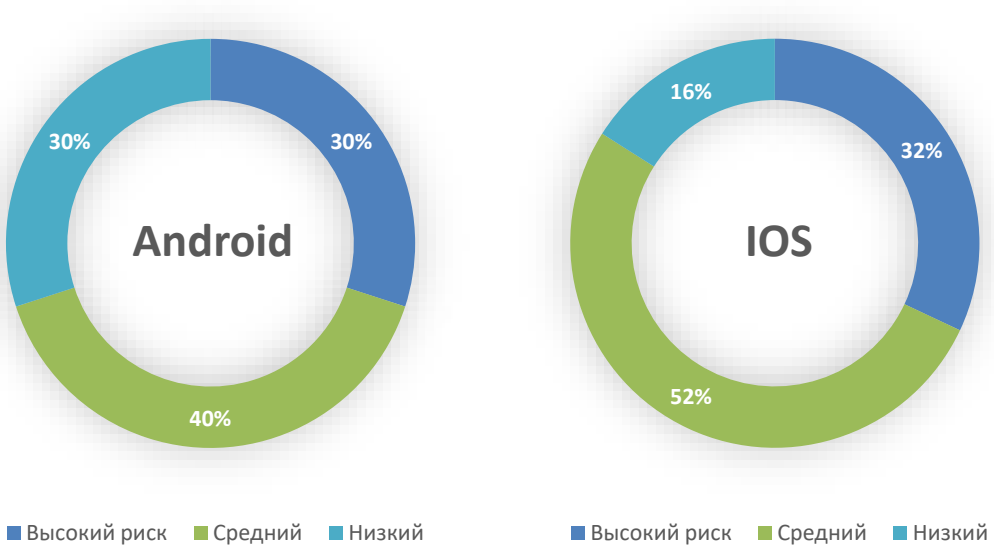


Рисунок 2

На рисунке 3 можно увидеть уровень защищенности клиентских частей (доля систем).

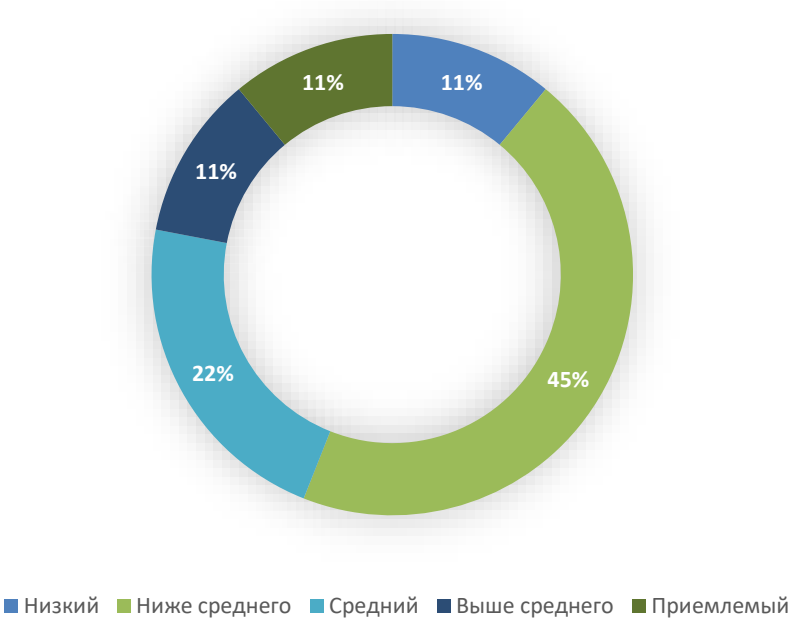


Рисунок 3

Уязвимости

На этапе проектирования интерфейсов взаимодействия компонентов приложения закладывается уязвимость небезопасного межпроцессного взаимодействия и относится к ошибкам в реализации механизмов защиты. Ошибки в механизмах защиты стали причиной 74% уязвимостей в приложениях для iOS и 57% уязвимостей для платформ Android (рисунок 4).

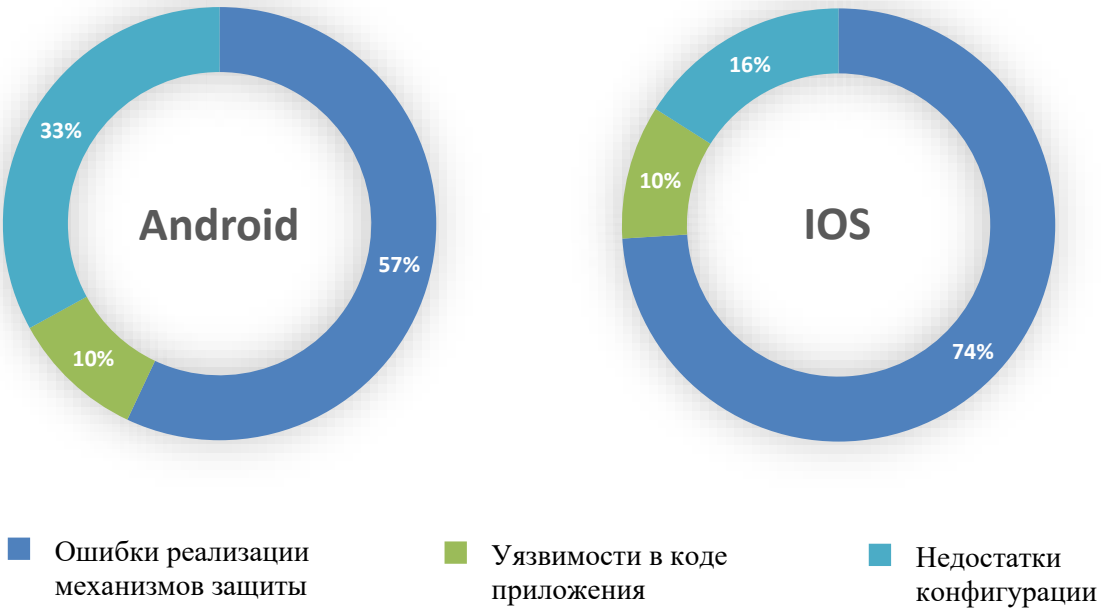


Рисунок 4

Примеры уязвимостей

- 25% приложений для платформ Android позволяют создавать резервную копию при подключении мобильного устройства к компьютеру
- 41% мобильных приложений осуществляют проверку аутентификационных данных на стороне клиента
- В августе 2018 года злоумышленники похитили персональные данные 20 000 пользователей мобильного приложения авиакомпании Air Canada
- 29% серверных частей содержат уязвимости, которые могут привести к нарушению работы приложения
- 18% приложений не ограничивают число попыток ввода аутентификационных данных
- Производитель виртуальной клавиатуры AI.type собирал чувствительные данные с мобильных устройств. Об этом стало известно после утечки базы данных 31 млн пользователей

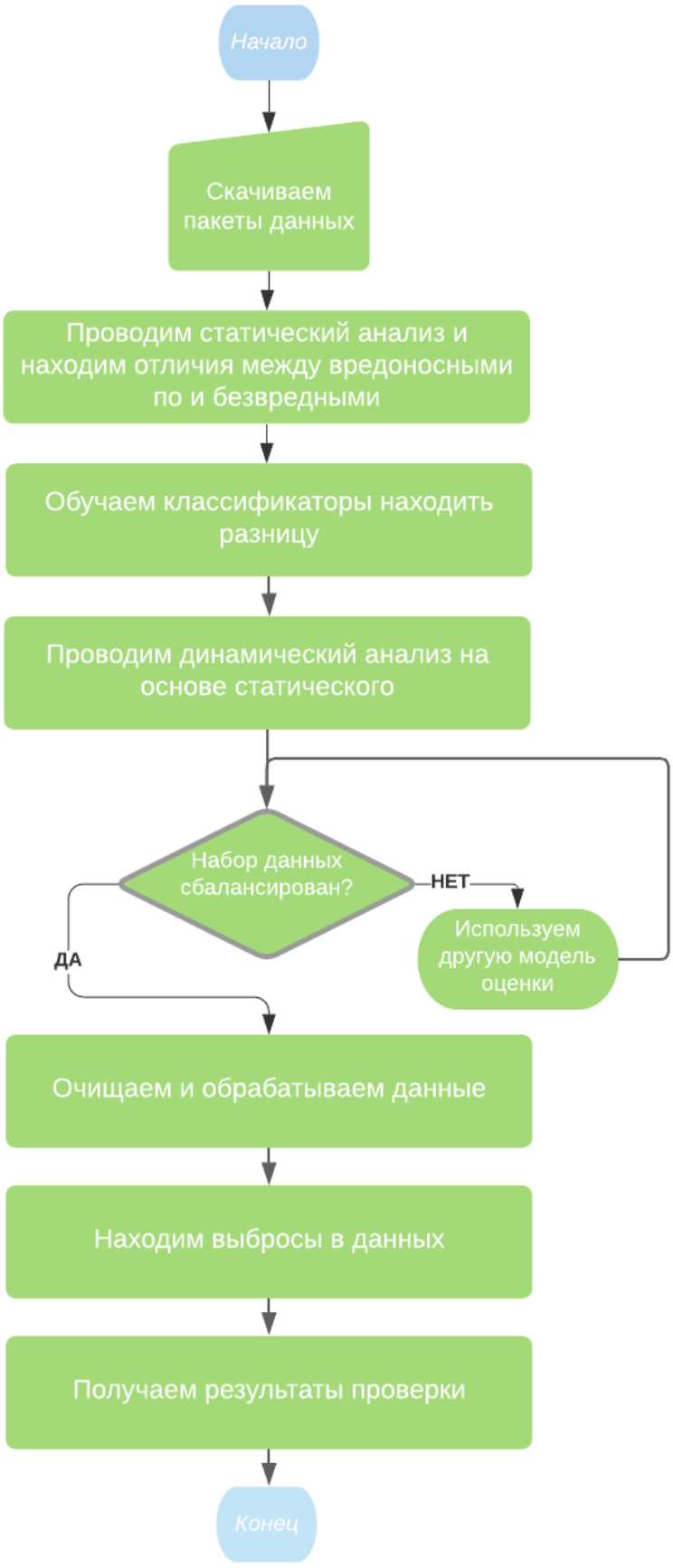
Моделирование объекта защиты

Для создания модели объекта защиты необходимо определить ту информацию, которую необходимо защищать. В данном случае это информация, которая хранится в телефонно-адресной книге. Проведем ее структурирование, путем классификации в соответствии с функциями, задачами.
К персональным данным телефонно-адресной книги относятся:

- фамилия, имя, отчество;
- год, месяц, дата рождения;
- номер телефона и адрес электронной почты;
- адрес проживания;
- имена близких;
- ссылки на соц. страницы;
- должность и место работы;
- фотография;
- номер банковского счета, паспортные данные и другая информация.

Все эти данные могут быть введены в телефонно-адресную книгу. Для того, чтобы проверить для чего приложение запрашивает доступ к телефонно-адресной книге (например, отправку личной информации на сторонние платформы) будет использоваться данный алгоритм (рисунок 5).

Рисунок 5


Начало
Скачиваем пакеты данных
Проводим статический анализ и находим отличия между вредоносными по и безвредными
Обучаем классификаторы находить разницу
Проводим динамический анализ на основе статического
Набор данных сбалансирован?
НЕТ
ДА
Используем другую модель оценки
Очищаем и обрабатываем данные
Находим выбросы в данных
Получаем результаты проверки
Конец

Программа получает входные данные и проверяет их на целостность. Проверка происходит с помощью алгоритма, в котором уже обученные классификаторы проверяют файл по заданным параметрам.
Классификаторы были обучены на пакетах данных, в которых были как безвредные, так и зловредные программы. Обучение происходило путем сравнения вредоносного ПО и доброкачественного. Из результатов были выведены критерии по проверке программы.
Далее, проверяется полученный результат проверки классификаторами на сбалансированность. Если он не сбалансирован, то мы меняем модель оценки, до тех пор, пока данные не станут сбалансированными, после чего происходит очистка и обработка. Затем для наглядности, переводим их в тепловые карты и находим выбросы, очищаем их от шума, после чего выводим результат проверки.

Разработка алгоритма

Статический анализ базы данных

Пакеты данных

In [2]:

```
from sklearn.naive_bayes import GaussianNB, BernoulliNB
from sklearn.metrics import accuracy_score, classification_report
from sklearn.ensemble import BaggingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import SGDClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import cohen_kappa_score
from sklearn.metrics import confusion_matrix
from sklearn.ensemble import RandomForestClassifier

from sklearn import preprocessing
import torch
from sklearn import svm
from sklearn import tree
import pandas as pd
from sklearn.externals import joblib
import pickle
import numpy as np
import seaborn as sns
```

```
/opt/conda/lib/python3.6/site-packages/sklearn/externals/joblib/__init__.py:
15: DeprecationWarning: sklearn.externals.joblib is deprecated in 0.21 and w
ill be removed in 0.23. Please import this functionality directly from jobli
b, which can be installed with: pip install joblib. If this warning is raise
d when loading pickled models, you may need to re-serialize those models wit
h scikit-learn 0.21+.
  warnings.warn(msg, category=DeprecationWarning)
```

Для дальнейшего анализа будет исследована база данных Malgenome

In [3]:
```
import pandas as pd
df = pd.read_csv("../input/datasetandroidpermissions/train.csv", sep=";")
```

In [4]:
```
df = df.astype("int64")
df.type.value_counts()
```

Out[4]:
```
1    199
0    199
Name: type, dtype: int64
```

Type - это метка, которая указывает, является ли приложение вредоносным или нет. В данном случае видно, что этот набор данных сбалансирован.

In [5]:
```
df.shape
```

Out[5]:
```
(398, 331)
```

Далее рассмотрим топ 10 решений, которые используются для образцов вредоносного ПО.

Вредоносное ПО

In [6]:
```
pd.Series.sort_values(df[df.type==1].sum(axis=0), ascending=False)[1:11]
```

Out[6]:
```
android.permission.INTERNET                195
android.permission.READ_PHONE_STATE        190
android.permission.ACCESS_NETWORK_STATE    167
android.permission.WRITE_EXTERNAL_STORAGE  136
android.permission.ACCESS_WIFI_STATE       135
android.permission.READ_SMS                124
android.permission.WRITE_SMS               104
android.permission.RECEIVE_BOOT_COMPLETED  102
android.permission.ACCESS_COARSE_LOCATION   80
android.permission.CHANGE_WIFI_STATE        75
dtype: int64
```

Безвредное ПО

In [7]:
```
pd.Series.sort_values(df[df.type==0].sum(axis=0), ascending=False)[:10]
```

Out[7]:
```
android.permission.INTERNET                 104
android.permission.WRITE_EXTERNAL_STORAGE    76
android.permission.ACCESS_NETWORK_STATE      62
android.permission.WAKE_LOCK                 36
android.permission.RECEIVE_BOOT_COMPLETED    30
android.permission.ACCESS_WIFI_STATE         29
android.permission.READ_PHONE_STATE          24
android.permission.VIBRATE                   21
android.permission.ACCESS_FINE_LOCATION      18
android.permission.READ_EXTERNAL_STORAGE     15
dtype: int64
```

In [8]:
```
import matplotlib.pyplot as plt
fig, axs =  plt.subplots(nrows=2, sharex=True)

pd.Series.sort_values(df[df.type==0].sum(axis=0), ascending=False)[:10].plot.ba
r(ax=axs[0])
pd.Series.sort_values(df[df.type==1].sum(axis=0), ascending=False)[1:11].plot.b
ar(ax=axs[1], color="red")
```

Out[8]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0198bb4390>
```

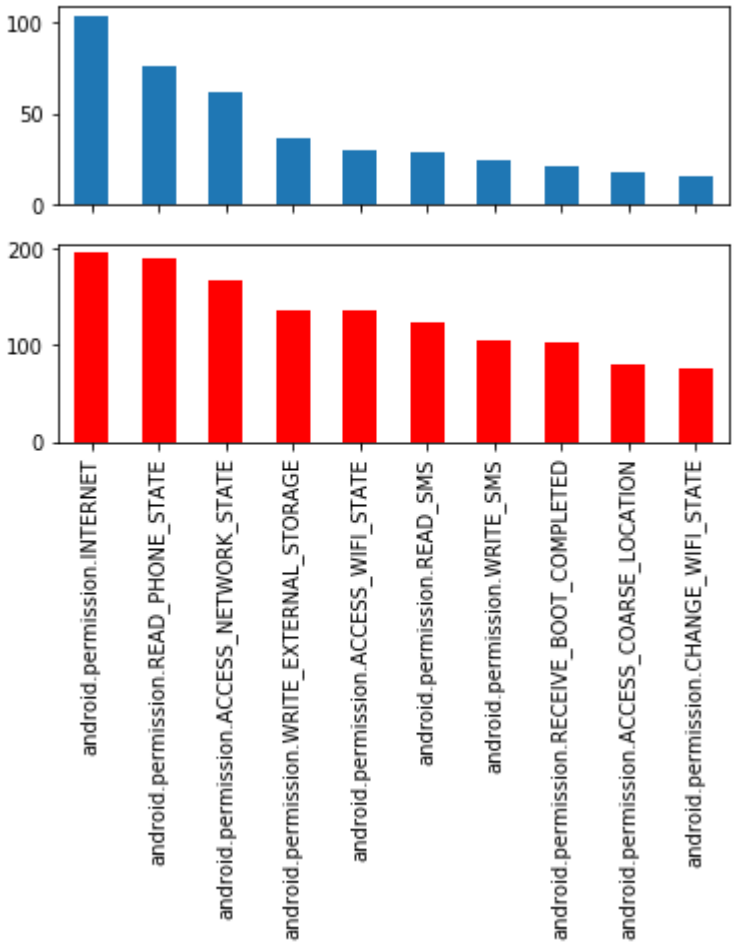


Рисунок 6

Результаты, выведенные в диаграмме (рисунок 6), позволяют понять разницу между разрешениями, используемыми вредоносным ПО и безобидными приложениями.

Обучение

Далее обучаем различные классификаторы обнаруживать вредоносное ПО. Будут использованы: метод k-ближайших соседей, дерево решений, наивный байесовский классификатор, но продемонстрирован только последний.

In [9]:
```
X_train, X_test, y_train, y_test = train_test_split(df.iloc[:, 1:330], df['typ
e'], test_size=0.20, random_state=42)
```

Наивный байесовский классификатор

In [10]:

```
# Naive Bayes algorithm
gnb = GaussianNB()
gnb.fit(X_train, y_train)

# pred
pred = gnb.predict(X_test)

# accuracy
accuracy = accuracy_score(pred, y_test)
print("naive_bayes")
print(accuracy)
print(classification_report(pred, y_test, labels=None))
```

```
naive_bayes
0.8375
              precision    recall  f1-score   support

           0       0.91      0.76      0.83        41
           1       0.78      0.92      0.85        39

    accuracy                           0.84        80
   macro avg       0.85      0.84      0.84        80
weighted avg       0.85      0.84      0.84        80
```

Динамический анализ

Для этого подхода был использован набор файлов pcap из проекта DroidCollector, где 4705 интегрированы доброкачественными и 7846 вредоносными приложениями. Все файлы были обработаны нашим сценарием извлечения признаков (результаты статического анализа). Согласно статическому анализу, во многих приложениях использовалось сетевое соединение, другими словами, приложения пытаются “общаться” и передавать информацию. Поэтому сейчас, с помощью динамического анализа будет проверено можно ли отличить вредоносное ПО от безобидного приложения, используя сетевой трафик.

In [13]:

```
import pandas as pd
data = pd.read_csv("../input/network-traffic-android-malware/android_traffic.csv", sep=";")
data.head()
```

Out[13]:

	name	tcp_packets	dist_port_tcp	external_ips	vulume_bytes	udp_packets	tcp_urg_packet
0	AntiVirus	36	6	3	3911	0	0
1	AntiVirus	117	0	9	23514	0	0
2	AntiVirus	196	0	6	24151	0	0
3	AntiVirus	6	0	1	889	0	0
4	AntiVirus	6	0	1	882	0	0

	source_app_packets	remote_app_packets	source_app_bytes	remote_app_bytes	duracion
0	39	33	5100	4140	NaN
1	128	107	26248	24358	NaN
2	205	214	163887	24867	NaN
3	7	6	819	975	NaN
4	7	6	819	968	NaN

	avg_local_pkt_rate	tavg_remote_pkt_rate	source_app_packets.1	dns_query_times	type
0	NaN	NaN	39	3	benign
1	NaN	NaN	128	11	benign
2	NaN	NaN	205	9	benign
3	NaN	NaN	7	1	benign
4	NaN	NaN	7	1	benign

In [14]:

```
data.columns
```

Out[14]:

```
Index(['name', 'tcp_packets', 'dist_port_tcp', 'external_ips', 'vulume_bytes',
       'udp_packets', 'tcp_urg_packet', 'source_app_packets',
       'remote_app_packets', 'source_app_bytes', 'remote_app_bytes',
       'duracion', 'avg_local_pkt_rate', 'avg_remote_pkt_rate',
       'source_app_packets.1', 'dns_query_times', 'type'],
      dtype='object')
```

In [15]:

```
data.shape
```

Out[15]:

```
(7845, 17)
```

In [16]:
```
data.type.value_counts()
```

Out[16]:
```
benign       4704
malicious    3141
Name: type, dtype: int64
```

В этом случае получаем несбалансированный набор данных, поэтому будет использоваться другая модель оценки.

Очистка и обработка данных

In [17]:
```
data.isna().sum()
```

Out[17]:
```
name                    0
tcp_packets             0
dist_port_tcp           0
external_ips            0
vulume_bytes            0
udp_packets             0
tcp_urg_packet          0
source_app_packets      0
remote_app_packets      0
source_app_bytes        0
remote_app_bytes        0
duracion             7845
avg_local_pkt_rate   7845
avg_remote_pkt_rate  7845
source_app_packets.1    0
dns_query_times         0
type                    0
dtype: int64
```

Когда происходила обработка каждого pcap, возникали проблемы с получением трех функций (продолжительность, средняя скорость удаленных пакетов, средняя скорость локальных пакетов). Как вы видите, сейчас этой проблемы нет.

In [18]:
```
data = data.drop(['duracion','avg_local_pkt_rate','avg_remote_pkt_rate'], axis=
1).copy()
```

In [19]:
```
data.describe()
```

Out[19]:

	tcp_packets	dist_port_tcp	external_ips	vulume_bytes	udp_packets	tcp_urg_packet
count	7845.000000	7845.000000	7845.000000	7.845000e+03	7845.000000	7845.000000
mean	147.578713	7.738177	2.748502	1.654375e+04	0.056724	0.000255
std	777.920084	51.654222	2.923005	8.225650e+04	1.394046	0.015966
min	0.000000	0.000000	0.000000	0.000000e+00	0.000000	0.000000
25%	6.000000	0.000000	1.000000	8.880000e+02	0.000000	0.000000
50%	25.000000	0.000000	2.000000	3.509000e+03	0.000000	0.000000
75%	93.000000	0.000000	4.000000	1.218900e+04	0.000000	0.000000
max	37143.000000	2167.000000	43.000000	4.226790e+06	65.000000	1.000000

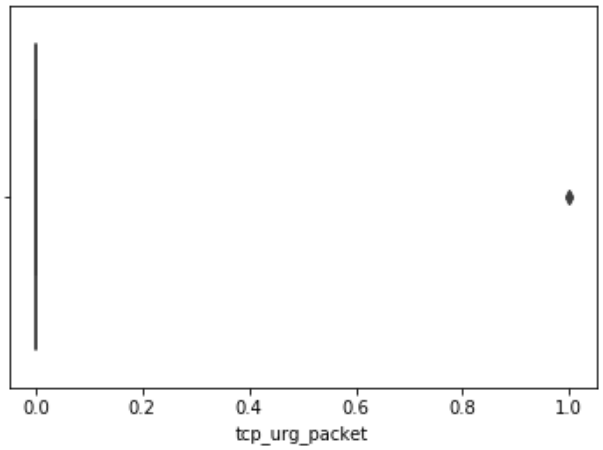
	source_app_packets	remote_app_packets	source_app_bytes	remote_app_bytes
count	7845.000000	7845.000000	7.845000e+03	7.845000e+03
mean	152.911918	194.706310	2.024967e+05	1.692260e+04
std	779.034618	1068.112696	1.401076e+06	8.238182e+04
min	1.000000	0.000000	0.000000e+00	6.900000e+01
25%	7.000000	7.000000	9.340000e+02	1.046000e+03
50%	30.000000	24.000000	4.090000e+03	3.803000e+03
75%	98.000000	92.000000	2.624400e+04	1.261000e+04
max	37150.000000	45928.000000	6.823516e+07	4.227323e+06

	source_app_packets.1	dns_query_times
count	7845.000000	7845.000000
mean	152.911918	4.898917
std	779.034618	18.900478
min	1.000000	0.000000
25%	7.000000	1.000000
50%	30.000000	3.000000
75%	98.000000	5.000000
max	37150.000000	913.000000

Теперь идея состоит в том, чтобы увидеть выбросы в данных.

In [20]:
```
sns.boxplot(data.tcp_urg_packet)
```

Out[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0198a0d828>
```



In [21]:
```
data.loc[data.tcp_urg_packet > 0].shape[0]
```

Out[21]:
```
2
```

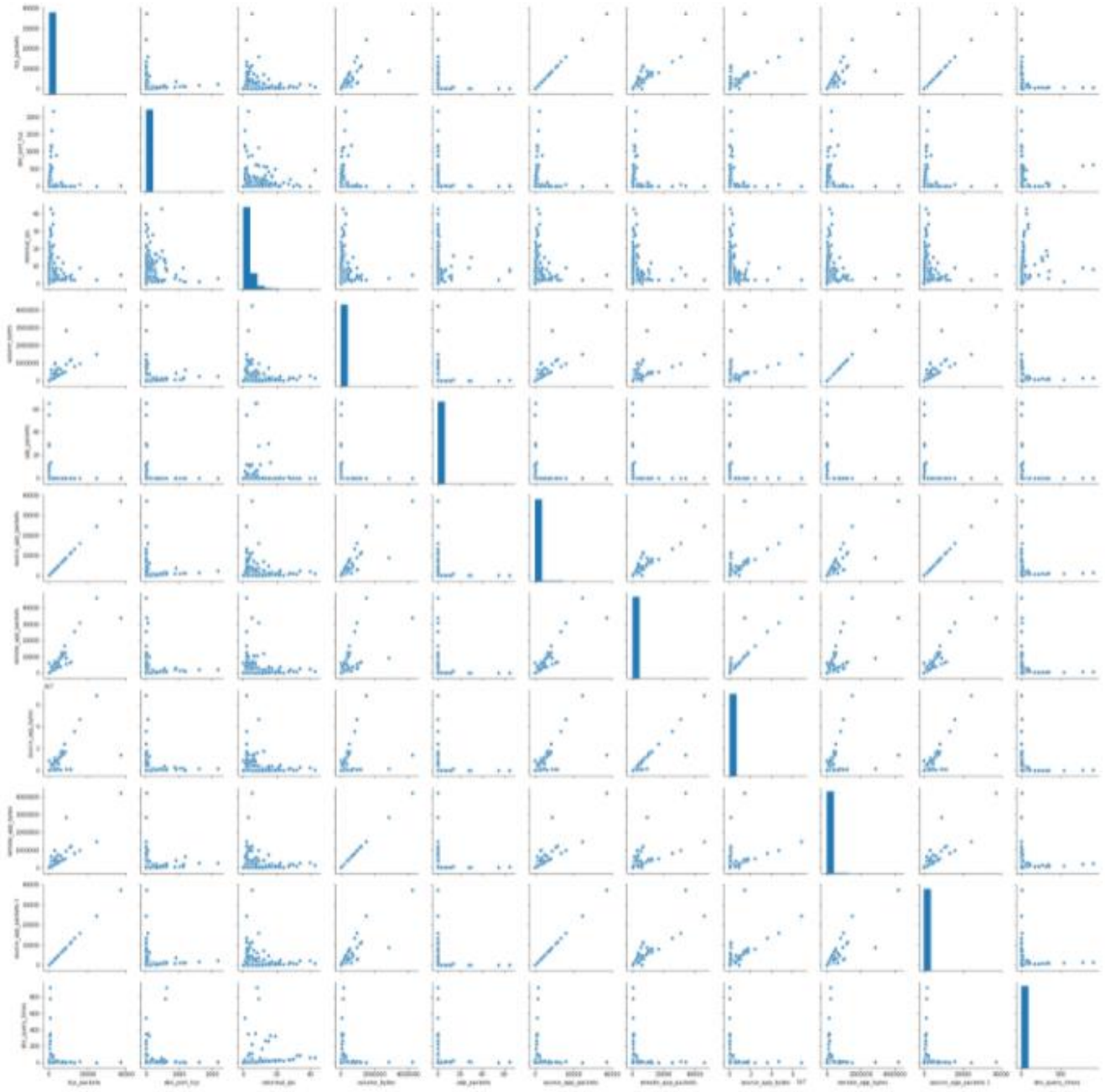
Этот столбец не будет использоваться для анализа, только две строки отличные от нуля, они будут важны для дальнейшего анализа.

In [22]:
```
data = data.drop(columns=["tcp_urg_packet"], axis=1).copy()
data.shape
```

Out[22]:
```
(7845, 13)
```

In [23]:
```
sns.pairplot(data)
```

Out[23]:
```
<seaborn.axisgrid.PairGrid at 0x7f0198aca908>
```



Можно заметить множество выбросов в некоторых функциях, опустим анализ глубины и получим только набор данных без шума.

In [24]:
```
data=data[data.tcp_packets<20000].copy()
data=data[data.dist_port_tcp<1400].copy()
data=data[data.external_ips<35].copy()
data=data[data.vulume_bytes<2000000].copy()
data=data[data.udp_packets<40].copy()
data=data[data.remote_app_packets<15000].copy()
```

In [25]:
```
data[data.duplicated()].sum()
```

Out[25]:
```
name                  AntiVirusAntiVirusAntiVirusAntiVirusAntiVirusA...
tcp_packets                                                       15038
dist_port_tcp                                                      3514
external_ips                                                       1434
vulume_bytes                                                    2061210
udp_packets                                                          38
source_app_packets                                                21720
remote_app_packets                                                18841
source_app_bytes                                                8615120
remote_app_bytes                                                2456160
source_app_packets.1                                              21720
dns_query_times                                                    5095
type                  benignbenignbenignbenignbenignbenignbenignbeni...
dtype: object
```

In [26]:
```
data=data.drop('source_app_packets.1',axis=1).copy()
```

In [27]:
```
scaler = preprocessing.RobustScaler()
scaledData = scaler.fit_transform(data.iloc[:,1:11])
scaledData = pd.DataFrame(scaledData, columns=['tcp_packets','dist_port_tcp','e
xternal_ips','vulume_bytes','udp_packets','source_app_packets','remote_app_pack
ets',' source_app_bytes','remote_app_bytes','dns_query_times'])
```

Из данного анализа придем к выводу, что лучшими сетевыми характеристиками являются:

TCP packets, это количество пакетов, отправленных и полученных TCP во время связи.

Dist port TCP, это общее количество пакетов, отличное от TCP.ф

External IP-адрес, представляет собой количество внешних адресов (IP-адресов), с которыми приложение пыталось установить связь.

Vulume_bytes, это количество байтов, которые были отправлены из приложения на внешние сайты.
UDP packets, общее количество пакетов UDP, переданных при обмене данными.
Source application packets- это количество пакетов, отправленных из приложения на удаленный сервер.
Remote app packets, количество пакетов, полученных из внешних источников.
Source application bytes, это объем (в байтах) связи между приложением и сервером.
Remote application bytes, это объем (в байтах) данных от сервера к эмулятору.
DNS-query, количество DNS-запросов.

Моделирование

In [28]:
```
X_train, X_test, y_train, y_test = train_test_split(scaledData.iloc[:,0:10], da
ta.type.astype("str"), test_size=0.25, random_state=45)
```

In [29]:
```
gnb = GaussianNB()
gnb.fit(X_train, y_train)
pred = gnb.predict(X_test)
## accuracy
accuracy = accuracy_score(y_test,pred)
print("naive_bayes")
print(accuracy)
print(classification_report(y_test,pred, labels=None))
print("cohen kappa score")
print(cohen_kappa_score(y_test, pred))
```

```
naive_bayes
0.44688457609805926
              precision    recall  f1-score   support

      benign       0.81      0.12      0.20      1190
   malicious       0.41      0.96      0.58       768

    accuracy                           0.45      1958
   macro avg       0.61      0.54      0.39      1958
weighted avg       0.66      0.45      0.35      1958

cohen kappa score
0.06082933470572538
```

In [30]:

```python
# kneighbors algorithm

for i in range(3,15,3):

    neigh = KNeighborsClassifier(n_neighbors=i)
    neigh.fit(X_train, y_train)
    pred = neigh.predict(X_test)
    # accuracy
    accuracy = accuracy_score(pred, y_test)
    print("kneighbors {}".format(i))
    print(accuracy)
    print(classification_report(pred, y_test, labels=None))
    print("cohen kappa score")
    print(cohen_kappa_score(y_test, pred))
    print("")
```

```
kneighbors 3
0.8861082737487231
              precision    recall  f1-score   support

      benign       0.90      0.91      0.91      1173
   malicious       0.87      0.85      0.86       785

    accuracy                           0.89      1958
   macro avg       0.88      0.88      0.88      1958
weighted avg       0.89      0.89      0.89      1958

cohen kappa score
0.7620541314671169
```

```
kneighbors 6
0.8784473953013279
              precision    recall  f1-score   support

      benign       0.92      0.88      0.90      1240
   malicious       0.81      0.87      0.84       718

    accuracy                           0.88      1958
   macro avg       0.87      0.88      0.87      1958
weighted avg       0.88      0.88      0.88      1958

cohen kappa score
0.7420746759356631

kneighbors 9
0.8707865168539326
              precision    recall  f1-score   support

      benign       0.89      0.90      0.89      1175
   malicious       0.85      0.83      0.84       783

    accuracy                           0.87      1958
   macro avg       0.87      0.86      0.86      1958
weighted avg       0.87      0.87      0.87      1958
```

```
kneighbors 12
0.8615934627170582
              precision    recall  f1-score   support

      benign       0.88      0.89      0.89      1185
   malicious       0.83      0.82      0.82       773

    accuracy                           0.86      1958
   macro avg       0.86      0.85      0.86      1958
weighted avg       0.86      0.86      0.86      1958

cohen kappa score
0.7100368862537227
```

In [31]:

```
rdF=RandomForestClassifier(n_estimators=250, max_depth=50,random_state=45)
rdF.fit(X_train,y_train)
pred=rdF.predict(X_test)
cm=confusion_matrix(y_test, pred)

accuracy = accuracy_score(y_test,pred)
print(rdF)
print(accuracy)
print(classification_report(y_test,pred, labels=None))
print("cohen kappa score")
print(cohen_kappa_score(y_test, pred))
print(cm)
```

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                       max_depth=50, max_features='auto', max_leaf_nodes=Non
e,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=250,
                       n_jobs=None, oob_score=False, random_state=45, verbos
e=0,
                       warm_start=False)
0.9172625127681308
              precision    recall  f1-score   support

      benign       0.93      0.94      0.93      1190
   malicious       0.90      0.88      0.89       768

    accuracy                           0.92      1958
   macro avg       0.91      0.91      0.91      1958
weighted avg       0.92      0.92      0.92      1958

cohen kappa score
0.8258206083396299
[[1117   73]
 [  89  679]]
```

Оценка эффективности модификации

Оценка качественных показателей
Ключевые особенности:

- Антивирусная защита в реальном времени — защита от вирусов, которая включается при запуске и непрерывно контролирует устройство, предотвращая выполнение зараженных файлов;
- Быстрые обновления доступны через GPRS вашего мобильного оператора или через подключенный к Интернету настольный компьютер. Благодаря постоянному обновлению вы защищены от вирусных эпидемий;
- Низкое потребление ресурсов — созданный специально для мобильных устройств, является «легким» инструментом, который использует оптимизированные процессы сканирования, чтобы сохранить заряд ваших батарей;
- Прост в установке и использовании — установка возможна непосредственно через мобильное интернет-подключение или через подключение к настольному компьютеру. Простой и дружественный интерфейс не требует особого вмешательства пользователя.

	Поддержка symbian	Поддержка Windows Mobile	Стоимость продукта, руб.	Частота обновлений	Русскоязычный интерфейс
система	да	да	0 руб	высокая	да

- Начальный уровень обнаружения новых, только что созданных вирусов составляет менее 5%. Хотя разработчики антивирусного ПО стараются постоянно обновлять свои методы обнаружения, первоначальный уровень обнаружения новых вирусов практически равен нулю. Исследователи считают, что большинство антивирусных продуктов на современном рынке не могут идти в ногу с темпами распространения вирусов в сети Интернет.
- Для некоторых разработчиков антивирусов необходимо до четырех недель, чтобы обнаружить новые вирусы с момента первоначального сканирования.
- Антивирусы с лучшими возможностями обнаружения, имеют также высокий процент ложных срабатываний.

ЗАКЛЮЧЕНИЕ

Этот набор данных является результатом исследований в области машинного обучения и безопасности Android. Данные были получены с помощью процесса, который заключался в создании двоичного вектора разрешений, используемых для каждого анализируемого приложения {1 = используется, 0 = не используется}.

Кроме того, образцы вредоносных / безвредных программ были разделены по типу; 1 вредоносное ПО и 0 не вредоносное ПО.

Теперь можно увидеть два подхода к анализу киберугроз. Конечно, можно использовать множество переменных, которые в данном случае не были использованы, например, потоки сети, вызовы методов, анализ графов и многие другие, но идея, лежащая в основе этой работы, состоит в том, чтобы понять, что нужно обращать внимание на все среды, потому что, работая в сфере кибербезопасности, можно столкнуться с комплексом сложных проблем.

Мобильные устройства — привлекательная мишень для хакерских атак, ведь в них хранятся и обрабатываются большие объемы личной информации и данные банковских карт. Как показывают результаты исследований, при разработке мобильных приложений вопросам безопасности уделяется недостаточно внимания, и основная проблема связана с небезопасным хранением данных. Хранение пользовательской информации в открытом виде, незащищенные данные на снимках состояния экрана, ключи и пароли в исходном коде — это лишь немногие из тех недостатков, что открывают просторы для проведения кибератак.

СПИСОК ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Стандарт ISO/IEC 27005:2018 "Information technology – Security techniques – Information security risk management" («Информационная технология. Методы и средства обеспечения безопасности. Менеджмент рисков информационной безопасности»).

2. Стандарт NIST SP 800-30 "Guide for Conducting Risk Assessments" («Руководство по проведению оценки рисков»).

3. Бельский В.С. Протокол обмена персональными данными: ИКС / В.С. Бельский, И.Ю. Герасимов, К.Д. Царегородцев, И.В. Чижов // International Journal of Open Information Technologies, 2020. – 23 с.

4. Urcuqui-López, C., & Cadavid, A. N. (2016). Framework for malware analysis in Android.

5. Urcuqui, C., Navarro, A., Osorio, J., & Garcıa, M. (2017). Machine Learning Classifiers to Detect Malicious Websites. CEUR Workshop Proceedings. Vol 1950, 14-17.