МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

Разработка программного приложения для восстановления повреждений PNG файлов

АВТОРЕФЕРАТ
дипломной работы

студента 6 курса 631 группы
специальности 10.05.01 Компьютерная безопасность
факультета компьютерных наук и информационных технологий
Шашкова Николая Владимировича

Научный руководитель
доцент, к.п.н. _____ А. С. Гераськин
19.01.2026 г.

Заведующий кафедрой
д. ф.-м. н., профессор _____ М. Б. Абросимов
19.01.2026 г.

Саратов 2026

ВВЕДЕНИЕ

Передача информации важнейшая составляющая гармонического и эффективного развития человечества. Без обмена данными не может в наши дни существовать ни одно научное сообщество. Но для того, чтобы совершить обмен, необходимо обладать эффективными способами сохранения информации. Одним из таких способов сохранения являются изображения.

Однако не существует способа сохранить информацию так, чтобы этот способ гарантировал неизменность сохранённых данных со временем, в результате чего имеем, что любые сохранённые данные имеют свойство искажаться под воздействием внешних факторов. Изображения не являются исключением из данного правила.

Если рассматривать различные форматы файлов изображений, то стоит отметить, что по данным W3Tech около 78.1% сайтов в интернете используют формат PNG.

Повреждения PNG файлов могут быть вызваны самыми различными сбоями. Это может быть, как ошибки при передаче данных или некорректное завершение записи, так и ошибки в программном обеспечении, которое непосредственно создаёт файлы такого формата. Также повреждения могут быть вызваны аппаратными сбоями или вредоносным программным обеспечением.

В данной работе будет рассмотрена структура PNG изображений, а также разработано приложение для восстановления повреждений PNG файлов.

Целью работы является разработка приложения для восстановления повреждений PNG файлов.

В соответствии с поставленной целью можно выделить следующие задачи:

1) Изучение структуры PNG файлов;

2) Изучение наиболее часто встречаемых структурных элементов PNG изображений;

3) Разработка программного приложения, способного восстанавливать структурные повреждения PNG изображения.

Дипломная работа состоит из введения, 4 разделов, заключения, списка использованных источников и 1 приложения. Общий объем работы – 72 страницы, из них 46 страниц – основное содержание, включая 25 рисунков и 27 таблиц, список использованных источников из 20 наименований.

КРАТКОЕ СОДЕРЖАНИЕ

Первый раздел дипломной работы содержит основную информацию о строении PNG изображений. В начале раздела приводится информация о структуре PNG изображения, описывается сигнатура PNG файла. Даётся информация о фрагментарной структуре PNG файлов. Также приводится информация о том, что каждый фрагмент состоит из как минимум 12 байт. Где первые 4 байта – размер фрагмента, вторые 4 байта – тип фрагмента, дальше следует некоторое количество байт данных фрагмента, последние 4 байта содержат информацию для проверки циклическим избыточным кодом. Структура фрагмента представлена в таблице 1.

Таблица 1 – Общая структура фрагментов

Length	Chunk Type	Chunk Data	CRC
4 байта	4 байта	Заданное в Length данного блока число байт	4 байта

Первый подраздел содержит информацию о критических фрагментах PNG. Описываются структура IHDR – фрагмента содержащего общую информацию об изображении, PLTE – фрагмента палитры изображения, IDAT – фрагмента фактических данных изображения и фрагмента IEND – отмечающего конец потока данных.

Второй подраздел содержит информацию о некритических фрагментах PNG изображения, в нём описываются публичные фрагменты PNG изображения с опорой на спецификации PNG.

В третьем подразделе первого раздела на выборке из 6000 изображений строится статистика встречаемости фрагментов PNG. Статистика представлена в таблице 2. Можно сделать вывод, что далеко не все описанные в спецификациях фрагменты применяются на практике. Приватные фрагменты применяются достаточно редко, однако, можно заметить, что примерно каждый 500 файл PNG изображения содержит в своём составе не описанный в спецификации фрагмент.

Таблица 2 – Статистика встречаемости фрагментов

Имя фрагмента	Количество	Среднее количество на 1 файл	Публичный или приватный
IDAT	82732	13.78867	Публичный
IHDR	6000	1.000000	Публичный
IEND	6000	1.000000	Публичный
PLTE	5026	0.837667	Публичный
cHRM	496	0.082667	Публичный
gAMA	493	0.082167	Публичный
pHYs	478	0.079667	Публичный
tEXt	397	0.066167	Публичный
bKGD	168	0.028000	Публичный
tIME	131	0.021833	Публичный
sRGB	64	0.010667	Публичный
tRNS	63	0.010500	Публичный
iCCP	50	0.008333	Публичный
iTXt	40	0.006667	Публичный
sBIT	27	0.004500	Публичный
eXIf	6	0.001000	Публичный
orNT	5	0.000833	Приватный
vpAg	4	0.000667	Приватный
zTXt	4	0.000667	Публичный
iDOT	2	0.000333	Приватный

Таким образом, можно сказать, что PNG файлы имеют блочную структуру, где критические фрагменты не должны иметь структурных повреждений, а остальные фрагменты должны иметь корректный Chunk Size. Наиболее важными элементами структуры PNG файлов являются Header, IHDR, IDAT, IEND. Также важен порядок расположения фрагментов файла. Для того чтобы изображение могло быть корректно отображено. Приватные фрагменты не столь часто встречаются, однако стоит отметить, что правила их именования не всегда соответствуют спецификации.

Во втором разделе рассматриваются приложения для восстановления PNG изображений. Рассматривается программа Kernel Photo Repair которая восстанавливает не сам файл изображения, а воссоздаёт изображение из

элементов, содержащих данные непосредственно изображения, таких, например, как IDAT, IHDR, PLTE. Данное приложение корректно восстанавливает изображение в случаях если:

1) Данные изображения не повреждены;
2) Структура основных фрагментов не повреждена;
3) Ни один из Chunk Length не повреждён.

Также рассматривается программный продукт iSunshare XRepair Genius, имеющий стандартный вариант восстановления изображения, восстанавливающий изображения не имеющие критичных повреждений структуры и расширенный вариант, который способен восстановить гораздо больший объём повреждений в случае если имеется неповреждённый файл схожей структуры. Данный функционал очень полезен при восстановлении PNG изображений, созданных с помощью одного и того же ресурса, поскольку в этом случае структуры PNG изображений будут очень схожи, что, вероятно, даёт расширенному алгоритму большую вероятность восстановления структуры.

В третьем разделе формулируется алгоритм восстановления структуры PNG изображения.

Также отмечается, что согласно спецификации PNG изображения мы всегда имеем представление о положении IHDR фрагмента, а также о наличии и наполнении IEND фрагмента. Затем приводится алгоритм определения степени соответствия входящего набора байт специфицированному фрагменту Вход: Набор байт предположительно являющийся фрагментом, фрагмент IHDR, число $0 < trustLevel < 1$

Выход: Список пар $(Chunk\ Type,\ tempN)$ с наиболее вероятными Chunk Type фрагментов

Шаг 1. Проверить длину размера байт, если она меньше чем 12, то вернуть в ответе пустой список и завершить алгоритм, иначе – перейти к шагу 2

Шаг 2. Разделить набор байт на 4 группы:

1) Length – первые 4 байта
2) Chunk type – вторые 4 байта
3) CRC – последние 4 байта
4) Data – все оставшиеся байты входного набора

Шаг 3. Сопоставить Data с шаблонами фрагментов с учётом данных из IHDR, для каждого фрагмента вычисляем $tempN$ – степень соответствия фрагмента шаблону.

Шаг 4. Для каждого фрагмента у которого $tempN \geq trustLevel$ внести $Chunk\ Type$ и $tempN$ в массив передаваемый на выход.

Шаг 5. Вернуть полученный на шаге 4 массив как результат работы алгоритма.

$tempN$ вычисляется по следующей логике:

1. Сравнить Длину Data и возможную длину данных шаблона, если длина Data – невозможна, то присвоить $tempN = 0$
2. Получить из шаблона число p – число структурных особенностей шаблона
3. Вычислить для Data число соответствующих им структурных особенностей, число $tempP$
4. Присвоить $tempN = tempP/p$

На основании данного алгоритма формулируется алгоритм восстановления фрагмента PNG изображения. Вход: Набор байт составляющих фрагмент, фрагмент IHDR файла из которого взят исследуемый набор байт, дробное число n – степень доверия, массив вхождений фрагментов.

Выход: Фрагмент с восстановленными метаданными (CRC, Chunk Type)

Шаг 1. Проверить длину размера байт, если она меньше чем 12, то вернуть null и завершить алгоритм, иначе – перейти к шагу 2

Шаг 2. Разделить набор байт на 4 группы:

1) Length – первые 4 байта
2) Chunk type – вторые 4 байта
3) CRC – последние 4 байта
4) Data – все оставшиеся байты входного набора

Шаг 3. Вычислить realCRC для Chunk Type и Data полученного набора байт если realCRC==CRC, то фрагмент не нуждается в восстановлении, иначе – перейти к шагу 4.

Шаг 4. Составить список $compareList = [(Chunk\ Type,\ tempN)]$ соответствия согласно алгоритму 1, а затем удалить из него фрагменты, которые не могут быть встречены на основании информации об уже встреченных фрагментах.

Шаг 5. Для каждого из фрагментов содержащихся в compareList вычислить d степень соответствия определённых в спецификации фрагментов. Сравнивая Chunk Type из входного набора байт с Chunk Type из массива Compare List и отсортировать по убыванию d полученный список. После этого выбрать элемент с наибольшим $tempN$ из тех, у кого наибольший равный d

Шаг 6. Заменить Chunk Type в полученном наборе байт на Chunk Type полученный на шаге 5. Пересчитать realCRC и записать его в значение CRC. Поместить в Length значение равное размерности Data

Шаг 7. Вернуть в результате работы изменённый набор байт.

Для восстановления структурных повреждений PNG изображения был разработан алгоритм 3 – восстановления структурных повреждений PNG изображения:

Шаг 1. Восстановить сигнатуру, Length и Chunk type фрагмента IHDR, полностью восстановить фрагмент IEND. Перейти к шагу 2.

Шаг 2. Создать: вспомогательные переменные: k, position, chunkSize, startPosition, объекты: previousChunk, thisChunk, newChunk, константу c = 12 (это размер блоков фрагмента Length, CRC, Chunk type в байтах), массив order

(порядок следования фрагментов), поместить в них значения фрагмента IHDR, перейти к шагу 3.

Шаг 3. Если $position < \text{размер файла} - 12$ то перейти к шагу 4, иначе перейти к шагу 12.

Шаг 4. Установить: $k = \text{индекс последнего элемета массива } order$, $previousChunk = \text{предпоследний занесённый в } order \text{ фрагмент}$(пустой в случае если в order всего один фрагмент) $lastChunk = \text{последний занесённый в } order \text{ чанк}$, $startPosition = lastChunk.startPosition$, $position = lastChunk.startPostition + lastChunk.size + c$ (тем самым мы получим номер байта изображения где предположительно начинается следующий фрагмент). Перейти к шагу 5.

Шаг 5. Попытаться на основании текущих данных восстановить заголовок у thisChunk. Перейти к шагу 6.

Шаг 6. Если фрагмент не является полностью корректным(position некорректно или CRC некорректно или через $position + lastChunk.size + 12$ от него не располагается фрагмент) перейти к шагу 7, иначе перейти к шагу 12

Шаг 7. Установить i = startPosition + c (начало данных обрабатываемого фрагмента), перейти к шагу 8.

Шаг 8. Если $i < \text{размер файла} - 12$ то перейти к шагу 9, иначе перейти к шагу 12

Шаг 9. Если на позиции i в файле начинается фрагмент то перейти к шагу 10, иначе перейти к шагу 11

Шаг 10. Изменить значение $size$ последнего фрагмента, установить $position = i$. Перейти к шагу 12

Шаг 11. $i = i + 1$, перейти к шагу 8.

Шаг 12. Записать в $lastChunk$ значение его данных. Если в файле встречался IEND перейти к шагу 13, иначе записать в newChunk значение фрагмента, начинающегося с байта position. Перейти к шагу 3

Шаг 13. В каждом фрагменте из order, size которого был повреждён, провести поиск фрагментов, которые могли быть им поглощены.

Шаг 14. Вернуть полученный файл в как результат работы, завершить алгоритм

Последний алгоритм реализует более комплексный подход, он учитывает порядок следования фрагментов, внутреннюю структуру фрагментов, для определения фрагмента используется следующие правила:

Набор встреченных данных длины n можно считать фрагментом PNG изображения если выполняется хотя бы одно из следующих условий:

1. Для встреченного фрагмента, его Chunk Data и Chunk Type соответствуют его CRC
2. Для встреченного фрагмента N можно найти фрагмент K определённый в спецификации, который соответствует следующим требованиям:
    a. Как минимум n из 4 байт или как минимум b из 32 бит Chunk Type совпадают у N фрагмента и K фрагмента. Где n и b выбираются в зависимости от требуемой точности. В рамках тестирования было выявлено, что наилучшие результаты получаются при $n = 3, b = 30$;
    b. Положение данного фрагмента относительно других фрагментов файла корректно, т.е. нет нарушения спецификаций, связанных с порядком встречаемости фрагментов и наличия/отсутствия взаимоисключающих фрагментов;
    c. Внутренняя структура фрагмента N во многом не противоречит фрагменту K.
3. Через $n + 12$ байт встречен поток байт произвольной длины k который может быть распознан как фрагмент

В четвертом разделе работы проводится демонстрация работы реализованного приложения, интерфейс которого представлен на рисунке 1. Приводятся примеры успешного и неуспешного вариантов срабатывания приложения. В работе приводятся следующие результаты тестирования приложения, реализующего алгоритмы, описанные в разделе 3.

С увеличением повреждения структуры снижается степень восстановления изображения. Путём дальнейшего тестирования алгоритма на изображениях различной степени повреждённости выявлено, что в основном программа справляется с восстановлением фрагментов, которые определены в спецификациях. Программа способна восстановить фрагмент, если в его структуре повреждено не более 5 байт, из которых не более 1 приходится на блок Chunk Type. Т.е. алгоритм способен восстановить PNG, если его структура повреждена в среднем на 39%

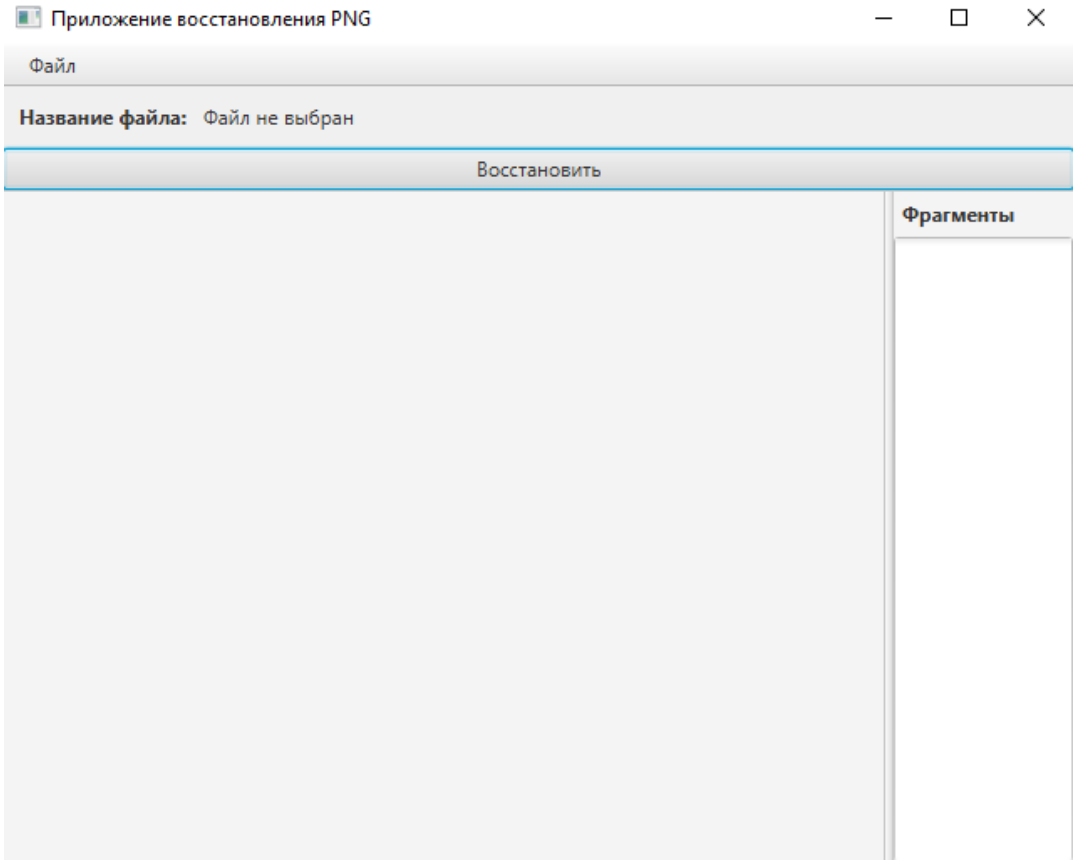

Рисунок 1 – Окно интерфейса приложения восстановления PNG

ЗАКЛЮЧЕНИЕ

В данной работе была рассмотрена и проанализирована структура PNG (Portable Network Graphics) изображений, разработано приложение для восстановления таких файлов.

Вероятность случайного повреждения структуры PNG зависит от соотношения количества фрагментов к объёму их содержимого. Чем меньше содержит каждый фрагмент, тем большая вероятность что случайная ошибка повредит именно структуру, а не содержимое изображения.

Наиболее часто встречаемые структурные элементы – критические фрагменты. Приватные фрагменты встречаются не очень часто, в среднем приходится порядка 1 приватного фрагмента на 500 PNG изображений.

В практической части работы был разработан и реализовано приложение, позволяющее восстанавливать повреждения структуры PNG изображений.

PNG формат имеет структуру, позволяющую восстанавливать её с применением программных средств, путём вычисления и анализа блоков Length и Chunk Type каждого последующего фрагмента, а также за счёт внутренней структуры каждого блока.

Алгоритм справляется повреждениями структуры, не превосходящими 39% от общего объёма структуры изображения. Причём в случае если изображение не содержит приватных фрагментов, то код восстанавливает порядка 98% таких файлов, в ином случае восстанавливает около 25% таких изображений.