Санкт-Петербургский государственный университет

Математическое обеспечение и администрирование информационных систем
Системное программирование

Медведев Андрей Александрович

Восстановление данных с дисков, повреждённых современными вредоносными программами

Выпускная квалификационная работа

Научный руководитель:
ст. пр. Губанов Ю. А.

Научный консультант:
ст. пр. Тимофеев Н. М.

Рецензент:
ст. пр. Ханов А. Р.

Санкт-Петербург
2018

SAINT-PETERSBURG STATE UNIVERSITY

Information Systems Administration and Mathematical Support
Software Engineering

Medvedev Andrei

Data recovery from disks damaged by modern malicious programs

Graduation Project

Scientific supervisor:
senior lecturer Gubanov Y. A.

Scientific consultant:
senior lecturer Timofeev N. M.

Reviewer:
senior lecturer Khanov A. R.

Saint-Petersburg
2018

Оглавление

Введение

Основной ценностью для пользователя компьютера являются его файлы – документы, фотографии, платёжные реквизиты, программы и т.д. Поэтому большинство современных операционных систем в первую очередь предназначены для упрощения работы с файлами.

Как и любой сложный программный продукт, операционная система неизбежно содержит дефекты. Этим пользуются злоумышленники, заинтересованные в использовании данных пользователя в собственных целях. Создавая вредоносные программы, они намеренно используют обнаруженные уязвимости и получают доступ к данным пользователя.

Одним из видов вредоносных программ являются программы-вымогатели. Общий цикл работы таких вредоносных программ после заражения компьютера делится на три этапа [5]. В первую очередь выполняется поиск целевых объектов программы: разделов на жёстком диске, файлов с конкретным расширением, файлов, содержащих в своём названии определённые ключевые слова, и т.п. Затем вредоносная программа блокирует доступ пользователя к обнаруженным объектам. После этого программа объявляет пользователю о том, что доступ к его данным заблокирован, и требует от него определённых действий для восстановления доступа. При этом никаких гарантий не предоставляется.

В случае, если вредоносная программа шифрует файл, чаще всего восстановить доступ к нему в обход требований вымогателя можно лишь расшифровав его. Однако, такой подход требует тщательного анализа алгоритма работы этой вредоносной программы. Поэтому такая задача решается индивидуально для каждой новой угрозы.

Если доступ к данным ограничивается шифрованием служебных структур или системных областей памяти, сами пользовательские данные остаются нетронутыми. Таким образом, доступ к данным может быть восстановлен, если удастся восстановить повреждённые служебные структуры.

Одним из примеров вредоносных программ, которые умышленно по-

вреждают служебные данные, является The Petya [9]. Несмотря на то, что непосредственно файлы на заражённом компьютере не модифицируются, доступ к ним полностью ограничивается из-за того, что вирус шифрует MFT-запись, главную файловую таблицу NTFS [16]. В MFT-записи содержится вся информация о файлах на томе файловой системы. The Petya предлагает пользователю перечислить денежные средства на счёт вымогателя и получить взамен код для расшифровки данных. Вместо того, чтобы выкупать ключ шифрования у злоумышленника, пользователь может попытаться восстановить файлы на повреждённом томе файловой системы и скопировать их на другой диск.

Belkasoft Evidence Center – инструмент цифрового криминалистического анализа [3], который позволяет анализировать и восстанавливать данные разных видов: файлы гибернации операционной системы, зашифрованные архивы, образы памяти мобильных устройств и т.д. Однако, при повреждении вредоносными программами служебных структур носителя информации, доступ к данным ограничивается, в связи с чем анализ данных усложняется. Для того чтобы восстановить доступ к данным необходимо в первую очередь восстановить повреждённые служебные структуры. Таким образом, представляет интерес задача восстановления доступа к данным на дисках с повреждёнными служебными структурами.

Постановка задачи

Целью данной работы является разработка программы для восстановления доступа к данным на дисках, повреждённых вредоносными программами. Для достижения поставленной цели были сформулированы следующие задачи:

- выполнить обзор служебных структур, уязвимых для атак вредоносными программами;
- исследовать способы восстановления доступа к данным при повреждении рассмотренных служебных структур;
- разработать архитектуру прототипа программы для восстановления доступа к данным;
- реализовать прототип программы;
- провести тестирование прототипа.

1. Обзор уязвимых служебных структур

1.1. MBR-запись

Главная загрузочная запись (Master Boot Record, MBR-запись), расположенная в первом секторе жёсткого диска, отвечает за управление разбиением жёсткого диска на разделы и передачу управления на загрузочный раздел при загрузке устройства [21].

С момента появления MBR-запись приобрела статус стандарта благодаря большой распространённости IBM PC-совместимых компьютеров. Однако, с развитием цифровых технологий и ростом требований к инструменту управления жёстким диском, MBR-запись потеряла свою актуальность из-за недостаточной надёжности и ограничений на размер разделов. В результате появился новый стандарт размещения таблиц разделов, GUID Partition Table (GPT), который считается лучшей альтернативой MBR-записи [19]. GPT-схема позволяет выделять гораздо больше пространства для разделов и обладает большей надёжностью за счёт хранения копий. Однако, MBR-запись до сих пор поддерживается и используется для совместимости со старыми устройствами.

1.1.1. Структура

Существует множество реализаций MBR-записи. В целях поддержания совместимости между ними обеспечивается совместимость с универсальной классической схемой. В ней содержится три основных компонента:

- код загрузчика;
- главная таблица разделов;
- сигнатура.

Код загрузчика. Исполняемый код загрузчика необходим процедуре загрузки компьютера для передачи управления на загрузочный раздел и последующей загрузки операционной системы. Разработчики за-

частую заменяют его вспомогательным трансферным кодом, который передаёт управление на другой участок памяти. Это позволяет фактически передавать управление любой программе перед загрузкой операционной системы. Таким образом, например, реализуются загрузчики, которые позволяют выбирать одну операционную систему из нескольких установленных на одном жёстком диске.

Исполняемый код загрузчика занимает первые 139 байт MBR-записи. После получения управления главная загрузочная запись находится в оперативной памяти по адресу 0000:7C00 и выполняет следующие инструкции.

1. Копирует загрузочный сектор из адреса 0000:7C00 по адресу 0000:0600, продолжает исполнение оттуда.

2. Проверяет главную таблицу разделов на наличие активных разделов.

3. Если существует единственный активный раздел, то копирует его первый сектор по адресу 0000:7C00, иначе, выводит ошибку.

4. Проверяет загрузочный сектор активного раздела на наличие сигнатуры (0xAA55).

5. Передаёт управление на загрузочный сектор активного раздела.

Во времена появления главной загрузочной записи оперативной памяти было очень мало, поэтому после загрузки первого сектора MBR-записи необходимо было оставить как можно больше свободного пространства для операционной системы. Загрузочный сектор занимает 512 байт, кроме того, может потребоваться дополнительное место для данных загрузочной программы, поэтому необходимо ещё 512 байт. Поскольку первую часть оперативной памяти процессоры, на которых впервые была применена MBR-запись (Intel 8086/8088), использовали для хранения векторов прерываний, под загрузочный сектор была выделена последняя часть первых 32 килобайт (минимальный размер оперативной памяти для работы DOS 1.0 [15]). Таким образом появился ад-

Таблица 1: Структура MBR-записи.

Смещение	Длина	Значение
0x00	1 байт	Признак активности раздела
0x01	3 байта	Начало раздела (CHS)
0x04	1 байт	Код файловой системы
0x05	3 байта	Конец раздела (CHS)
0x08	4 байта	Начало раздела (LBA)
0x0C	4 байта	Количество секторов раздела

рес 0000:7C00, обозначающий последние 1024 байт первых 32 килобайт оперативной памяти [23]. Именно по этому адресу загружается MBR в оперативную память.

Сразу за исполняемым кодом следуют сообщения ошибок, которые поддерживает классическая схема MBR.

Главная таблица разделов. Главная таблица разделов состоит не более чем из четырёх записей, каждая из которых отвечает за раздел на жёстком диске. Записи, содержащиеся в главной таблице разделов, называются первичными. Если организация данных на диске требует более чем четырёх разделов, то в таблице разделов допускается запись, хранящая указатель на расширенный раздел. Записи в расширенном разделе называются вторичными.

Каждая запись представляет из себя структуру длиной 16 байт. В этой структуре содержится информация об активности раздела, его местонахождении и формате файловой системы.

Признак активности раздела служит для определения раздела, с которого следует продолжить дальнейший процесс загрузки. Значение 0x80 обозначает активный раздел, а 0x00 – неактивный. Признаком активного раздела может обладать только первичный раздел.

Существует два основных способа адресации пространства на жёстком диске: цилиндр-головка-сектор (Cylinder Head Sector, CHS) [18] и логическая адресация (Logical Block Adressing, LBA) [20]. CHS основывается на физической геометрии диска и для адресации использует

номер цилиндра, номер сектора и позицию головки, в то время как LBA использует логическую адресацию.

Запись в таблице разделов поддерживает несколько видов адресаций: CHS и LBA. Поддержка LBA была введена для возможности создания разделов размером больше 7.8 гигабайт. Ограничение на максимальный размер раздела в два терабайта при размере сектора 512 байт существует из-за того, что длина раздела в секторах содержится в четырёх байтах.

Код файловой системы указывает на формат файловой системы раздела. Кроме того, существуют специальные значения, которые позволяют определить указанный раздел как служебный участок памяти, например, расширенный раздел.

Сигнатура. В двух последних байтах MBR-записи должна находиться специальная сигнатура: 0xAA55. Отсутствие сигнатуры указывает на то, что MBR-запись отсутствует или повреждена.

1.1.2. Роль в процедуре загрузки компьютера

В классической схеме загрузки компьютера после успешного завершения процедуры самотестирования (Power-On Self Test, POST) базовая система ввода-вывода (BIOS) пытается найти MBR-запись, расположенную в самом первом секторе на одном из устройств в очереди загрузки, определённой в BIOS. Как только главная загрузочная запись найдена, она копируется в оперативную память по адресу 0000:7C00, и затем BIOS передаёт управление на скопированный участок памяти.

1.1.3. Уязвимость

Как видно из описания внутренней структуры MBR-записи, она не имеет внутреннего механизма защиты. Поэтому, если вредоносной программе удастся заменить MBR-запись на свой исполняемый код, то при следующей загрузке устройства она получит полный контроль над устройством до загрузки операционной системы.

Существует много вредоносных программ, которые заменяют MBR-запись для получения доступа к пользовательским данным. Их называют Bootkit [4]. Ключевой особенностью вредоносных программ такого типа является сложность их обнаружения, поскольку их компоненты находятся вне файловой системы. Меры противодействия могут привести к нарушению работы компьютера из-за повреждения MBR-записи. Наиболее известными из семейств подобных вредоносных программ ыявляются семейства Stoned и Alureon.

1.2. MFT-таблица

Главная файловая таблица (Master File Table, MFT-таблица) содержит записи о каждом файле на томе файловой системы NTFS [16]. MFT-таблица размещается в специально выделенной MFT-зоне, которая по стандарту занимает около 12.5% свободного места при форматировании. При необходимости она может сокращаться, но файловая система избегает таких мер, поскольку дальнейший рост MFT-таблицы может привести к её фрагментации и понижению производительности работы файловой системы.

Файловая система NTFS для удобства адресации памяти делит её на секторы. Размер сектора может варьироваться от 512 байт до 64 килобайт, но стандартным размером считается четыре килобайта. Для хранения файлов NTFS выделяет не занятую MFT-зоной область памяти.

Для управления адресным пространством NTFS-раздела используются метафайлы. В этих недоступных для записи из операционной системы файлах содержится служебная информация о состоянии тома, доступном для записи пространстве и т.д.

1.2.1. Структура

Главная файловая таблица представляет из себя централизованный каталог, включающий в себя информацию обо всех файлах NTFS-раздела, в том числе о служебных файлах. Информация о файле содержится в

Таблица 2: Основные значения заголовка MFT-записи

Смещение	Длина	Значение
0x00	4 байта	Сигнатура состояния файла
0x20	2 байта	Смещение первого аттрибута
0x22	2 байта	Флаги состояния файла
0x28	4 байта	Физический размер MFT-записи

соответствующей записи таблицы. Каждая файловая запись в таблице занимает по умолчанию один килобайт, хотя на дисках с большим размером сектора размер может отличаться [7], его значение можно определить по значению переменной в загрузочном секторе.

Каждая запись в MFT-таблице состоит из заголовка и атрибутов файла. Уникальной особенностью файловой системы NTFS является возможность хранить файлы небольшого размера непосредственно в записи MFT-таблицы в качестве одного из атрибутов.

Информация о содержимом записи MFT-таблицы содержится в заголовке. В нём определена сигнатура файла, флаги, определяющие состояние файла, смещение первого атрибута и другая служебная информация [12]. Смещения и размер основных значений заголовка MFT-записи представлены в (Таб. 2).

Атрибут MFT-записи содержит информацию определённого типа о соответствующем файле [10]. Так, например, атрибут FileName содержит информацию о названии файла, а атрибут DataAttribute содержит информацию о том, как файл хранится в физической памяти.

1.2.2. Уязвимость

У MFT-таблицы существует частичная копия, называемая MFT Mirror. Она содержит в себе записи о метафайлах NTFS. Эта копия позволяет восстановить MFT-таблицу встроенным стандартным приложением chkdsk [11], но лишь при небольших повреждениях, таких как удаление нескольких первых записей в таблице. Однако, если главная файловая таблица сильно повреждена или MFT Mirror недоступен, то доступ к

данным не может быть восстановлен стандартными средствами.

Одной из вредоносных программ, которая блокирует доступ пользователя к данным посредством шифрования MFT-таблицы является The Petya [9]. Первая волна заражений компьютеров The Petya была зафиксирована весной 2016 года. Деятельность этой программы на тот момент ограничивалась внедрением в MBR-запись и шифрованием MFT-таблицы с последующим предложением выкупа ключей шифрования. Несмотря на то, что непосредственно шифрованию подвержен небольшой участок памяти, пользователь теряет доступ ко всему тому раздела.

2. Способы восстановления доступа к данным

В данной главе на основании внутреннего устройства MBR-записи и MFT-таблицы (гл. 1) проводится исследование способов восстановления доступа к данным при их повреждении.

2.1. Восстановление MBR-записи

В ситуации, когда на устройстве уничтожена главная загрузочная запись, компьютер не сможет завершить процесс загрузки, потому что ключевой элемент этой процедуры был утерян. Если остальные данные остались целыми, задача восстановления доступа к данным сводится к задаче воссоздания таблицы разделов. Код загрузчика можно временно заменить классическим для того, чтобы при необходимости получить возможность воспользоваться устройством хранения данных.

Чтобы восстановить таблицу разделов, необходимо восстановить информацию о расположении разделов на устройстве: начало, размер и формат файловой системы. Для этого необходимо исследовать вероятные места хранения разделов на устройстве в поисках типичных для файловых систем сигнатур или служебных структур.

Организация данных на запоминающем устройстве под управлением MBR-записи выглядит следующим образом: в первых нескольких секторах физической памяти располагаются служебные сектора, в том числе и главная загрузочная запись, далее следуют размеченные разделы.

Обычно под MBR-запись и служебные структуры резервируется нулевой цилиндр в CHS-адресации. Поэтому первым доступным для записи адресом в формате CHS является адрес первой головки на первом секторе для второго цилиндра. В формате LBA это 63-й сектор. Некоторые операционные системы использовали этот сектор для расположения системного раздела, например, Windows XP [6].

Не существует строгих правил распределения или выравнивания ад-

ресов начала разделов в памяти. Однако, существуют некоторые общепринятые практики, которые в некоторых случаях позволяют получить увеличение производительности команд обмена памятью. Одно из таких правил заключается в выравнивании первого сектора раздела относительно адреса, кратного четырём килобайтам. Такой подход может увеличить производительность для дисков, в которых внутренний размер сектора отличается от стандартного значения 512 байт и равен четырём килобайтам. Для программного обеспечения эмулируется режим обращения к 512 байтам, хотя на самом деле обрабатывается объём данных в восемь раз больше. Для большинства современных ОС это не является проблемой, поскольку в них самих по умолчанию большинство операций с носителем информации проходят порциями в четыре килобайта, но если физические границы секторов по четыре килобайта не совпадают с логическими, то одна операция записи или чтения блока длиной в четыре килобайта может выполняться как несколько, что заметно отразится на производительности. Поэтому, адреса начала разделов вероятнее кратны четырём килобайтам [22].

Для управления данными на разделах существуют файловые системы, которые используются для хранения, управления и изменения данных. Известно много различных файловых систем каждая из которых уникальна по своей структуре и обладает своими служебными участками памяти. Поэтому универсального средства для определения начала файловой системы нет. Однако, существует подход, благодаря которому можно попытаться определить начало раздела, содержащего файловую систему.

В классической схеме MBR одним из действий которое выполняет загрузочный код при исполнении, является проверка найденного активного раздела по специальной сигнатуре (0xAA55) в конце сегмента. Поэтому велика вероятность того, что в начале раздела будет расположен сектор, содержащий такую сигнатуру в последних двух байтах.

Кроме того, ключевые структуры большинства файловых систем детально известны. Так, для большинства UNIX-совместимых файловых систем характерно использование таких структур как superblock, inode

и т.д. В процессе исследования участка памяти на содержание одной из таких структур можно попытаться сопоставить имеющиеся данные со строением предполагаемой структуры для определения принадлежности к определённому типу файловой системы. Некоторые файловые системы используют характерные для них значения, например, файловая система NTFS содержит в первых нескольких байтах инструкцию для перехода на исполняемый код и сигнатуру 0x205346544E ('NTFS ').

В результате, восстановить таблицу разделов можно, исследовав физическую память на наличие характерных сигнатур и структур файловых систем для определения расположения раздела и создав соответствующую запись в новой таблице разделов. Искать их следует прежде всего по адресам кратным четырём килобайтам и в 63 секторе LBA-адресации.

2.2. Восстановление MFT-таблицы

MFT-таблица может быть восстановлена стандартными средствами, такими как chkdsk, в случае если она была частично повреждена и присутствует MFT Mirror. В ином случае, восстановить MFT-таблицу становится практически невозможно, потому что информация, которая хранилась в этой таблице, может больше нигде не встречаться. По этой причине при критическом повреждении MFT-таблиц средства восстановления данных зачастую восстанавливают файлы без их названий, поскольку имена файлов хранятся в главной файловой таблице соответствующего тома.

Поскольку главная файловая таблица является единственным источником знаний о расположении файлов на томе раздела, при её утрате остаётся лишь большой неразмеченный участок памяти в котором расположены нетронутые файлы. Стандартными средствами установить расположение конкретного файла невозможно.

Однако, существует метод сигнатурного поиска, благодаря которому можно попытаться найти все файлы интересующего формата и сохранить их на другой носитель. Этот метод основывается на том прин-

ципе, что программе заранее известна внутренняя структура файла необходимого формата и она может определить принадлежность участка памяти файлу. Таким образом, программа способна скопировать из образа повреждённого тома файлы нужного формата.

Несмотря на то, что внутренняя структура файлов конкретного формата заранее известна, метод сигнатурного поиска не всегда обеспечивает желаемый результат, поскольку файлы на томе могут быть фрагментированы. Это происходит из-за того, что в файловой системе не нашлось сплошного участка памяти для хранения файла и он был разбит на несколько частей. Некоторые средства восстановления данных используют метод сигнатурного поиска который игнорирует фрагментацию и при нахождении известных заголовков файлов просто копируют участок памяти необходимой длины. Поэтому в полученном файле находится лишь часть необходимых данных.

Пользуясь вышеизложенной информацией восстановить доступ к данным на томе с уничтоженной MFT-таблицей можно использовав метод сигнатурного поиска для файлов интересующего формата с учётом их возможной фрагментации.

3. Архитектура прототипа

3.1. Требования

Были выделены основные требования. Инструмент восстановления данных разрабатывается с целью дальнейшей апробации в продукте Belkasoft Evidence Center, поэтому он должен соответствовать основным требованиям продукта. Целевой операционной системой является ОС Windows. В качестве языка программирования выбран C#, поскольку этот язык программирования используется для реализации Belkasoft Evidence Center.

В соответствии с задачами восстановления MBR-записи и MFT-таблицы программа должна предоставлять следующую функциональность:

- восстановление главной загрузочной записи на образе жёсткого диска;
- восстановление файлов с тома, содержащего повреждённую главную таблицу файлов:
  - выбор интересующих пользователя форматов файлов для восстановления;
  - выбор интересующих пользователя файлов для сохранения.

Для восстановления MBR-записи на образе жёсткого диска в прототипе решено поддержать файловые системы NTFS и FAT32. Эти файловые системы используются по умолчанию для форматирования разделов на устройствах под управлением целевой операционной системы Windows.

Для восстановления данных с диска при повреждённой MFT-таблице решено поддержать формат файлов JPEG. Этот формат используется для хранения цифровых изображений и фотографий. В ситуации, когда злоумышленник избавляясь от доказательств его причастности к преступлению удаляет фотографию с устройства, специалист может вос-

пользоваться реализуемым инструментом для восстановления этой фотографии. Это становится возможным благодаря тому, что при обычном удалении файла с магнитного диска уничтожается лишь запись о нём в главной файловой таблице, а данные остаются на устройстве до перезаписи файловой системой. При этом JPEG сочетает в себе сложные структурированные данные и сплошной байтовый поток, что позволит оценить сигнатурный способ восстановления данных с разных точек зрения.

3.2. Компоненты

Прототип решено спроектировать из нескольких компонентов, каждый из которых отвечает за определённую задачу восстановления данных. За восстановление MBR-записи отвечает компонент MBRRestore, за восстановление данных при повреждённой MFT-таблице – MFTRestore. Взаимодействие с описанными компонентами осуществляется через пользовательский интерфейс. Диаграмма компонентов изображена на Рис. 1.

Используя графический интерфейс пользователь указывает путь к образу жёсткого диска, необходимый способ восстановления данных и интересующие форматы файлов при необходимости. Далее выбранный компонент начинает работу по восстановлению данных в соответствии с указанными пользователем параметрами.

Образ жёсткого диска находится под управлением класса ImageHolder, который определяет доступные для чтения участки памяти. Поскольку процесс восстановления данных должен учитывать деление памяти на

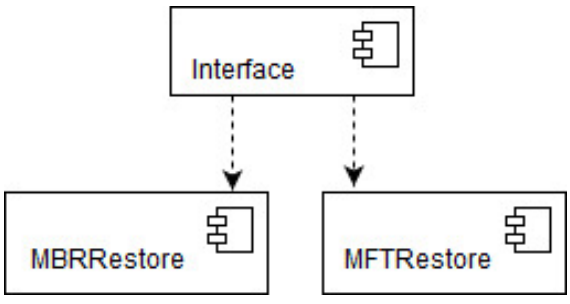

Рис. 1: Компоненты программы

секторы, обработка участков памяти должна быть организована как поэтапный анализ секторов на жёстком диске. Для этого определён класс Cluster, хранящий в себе участок памяти, соответствующий обрабатываемому сектору.

MBRRestore. MBRRestore восстанавливает главную загрузочную таблицу и создаёт новую MBR-запись, позволяющую получить доступ к данным. Для классификации участков памяти по принадлежности к одной из поддерживаемых файловых систем определён абстрактный класс FileSystemCarver. Соответственно, за идентификацию файловой системы NTFS отвечает класс NTFSCarver, за идентификацию FAT32 – FAT32Carver.

MFTRestore. MFTRestore восстанавливает файлы необходимого формата с учётом их возможной фрагментации и сохраняет успешно восстановленные файлы.

Для каждого поддерживаемого формата файлов определён класс, отвечающий за восстановление файлов соответствующего формата и реализующий абстрактный класс FormatCarver. За восстановление файлов формата JPEG отвечает класс JPEGCarver. Для определения наличия в исследуемом секторе части файла соответствующего формата и создания задачи восстановления используется абстрактный метод TryCreateTask класса FormatCarver. Каждая реализация класса FormatCarver по-своему определяет процесс идентификации сектора.

В случае, если восстанавливаемый файл оказался фрагментирован, необходимо приостановить работу над ним и попытаться возобновить её позднее на другом участке памяти. Для этого все необходимые данные сохраняются в реализации абстрактного класса Task, соответствующей формату файла. Task определяет метод Verify, который позволяет определить: возможно ли продолжить процесс восстановления файла из указанного сектора. Для формата JPEG определён класс-хранилище JPEGTask.

Диаграмма классов, отвечающих за процесс восстановления файлов

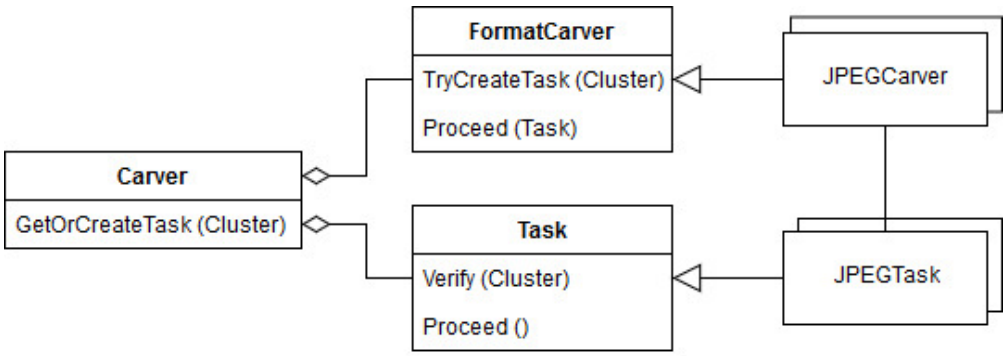

Рис. 2: Классы компонента ”MFTRestore”

разных форматов изображена на Рис. 2. После завершения процедуры восстановления данных пользователь выбирает интересующие его файлы для сохранения, остальные файлы удаляются.

4. Особенности реализации прототипа

4.1. Компонент восстановления MBR-записи

Для восстановления доступа к данным на жёстком диске с повреждённой MBR-записью достаточно заменить повреждённые сектора новой MBR-записью с классическим кодом загрузчика и соответствующей таблицей разделов диска.

Процедура восстановления главной загрузочной таблицы представляет из себя цикл, в котором каждый сектор образа исследуется на соответствие сигнатурам известных файловых систем. Если сектор удовлетворяет сигнатуре некоторой файловой системы, то компонент проверяет следущие сектора на соответствие внутренней структуре определённой файловой системы. В случае, если проверка пройдена успешно, инструмент устанавливает длину раздела, записывает соответствующую информацию в таблицу разбиений и продолжает цикл после последнего сектора найденного раздела.

Поскольку в главной загрузочной таблице существует ограничение на количество записей, может потребоваться создать запись, указывающую на следующую, второстепенную таблицу разделов. В дальнейшем заполняться будет именно второстепенная таблица.

По завершении обхода образа диска загрузочный код классической схемы записывается в первый сектор новой MBR-записи. Далее, полученная MBR-запись записывается в начало образа диска.

4.2. Компонент восстановления данных при повреждённой MFT-таблице

При повреждённой MFT-таблице восстановление данных производится с помощью метода сигнатурного поиска.

Процесс восстановления представляет из себя серию обходов образа жёсткого диска. При первом обходе создаются задачи по восстановлению файлов выбранных пользователем форматов. В процессе следующих обходов ранее созданные задачи продолжают выполняться на

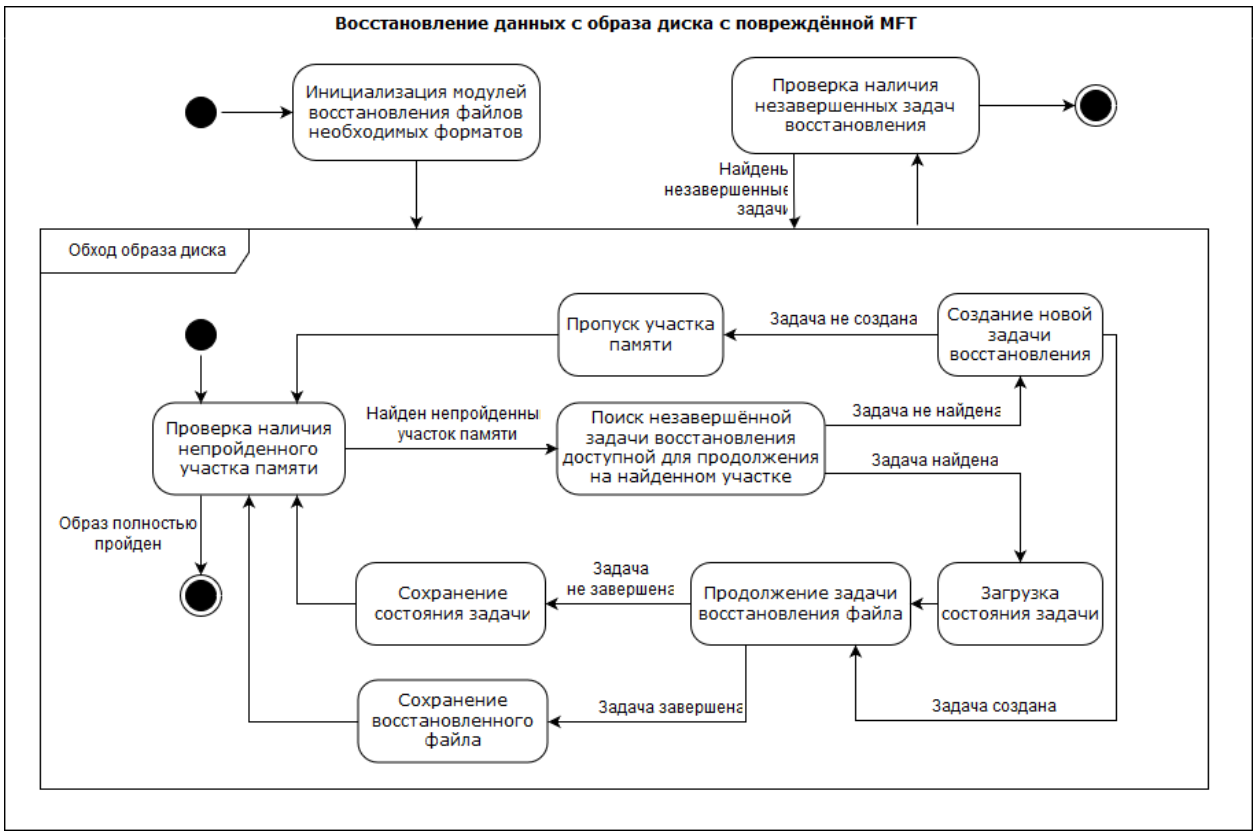

Рис. 3: Состояния компонента ”MFTRestore”

секторах, которые являются для них наиболее подходящими. Обходы образа жёсткого диска повторяются до тех пор, пока все задачи не завершатся, при этом ранее использованные сектора в обходе не участвуют. Диаграмма состояний изображена на (Рис. 3).

При обнаружении фрагментированного файла Carver пытается определить: подходит ли следующий сектор для продолжения текущей задачи восстановления.

Если задача была прервана на структуризованном участке файла и сектор удовлетворяет ожидаемой структуре, то задача продолжает исполняться, иначе, состояние задачи сохраняется, а прототип предпринимает попытку продолжить другую задачу восстановления или создать новую со следующего сектора. Если реализациями FormatCarver, соответствующими выбранным форматам файлов, сектор не был распознан, то он игнорируется на время текущего обхода.

Если задача восстановления фрагментированного файла прервалась на неструктуризованном потоке байтов, то для определения следующе-

го сектора необходимо установить способ, с помощью которого можно будет сравнить два сектора. В контексте восстановления JPEG-файла можно предположить, что соседние участки изображения не должны сильно отличаться друг от друга по цвету на смыкающихся границах. Поэтому для того, чтобы определить следующий подходящий сектор, необходимо сравнивать значения цветов на границах кадров изображения.

В качестве одного из способов сравнения секторов можно использовать усреднённую разницу в коэффициентах цветовой модели RGB (Red Green Blue). Но у такого подхода есть несколько недостатков. Во-первых, изменение цвета в модели RGB нелинейно, поскольку цветовые компоненты этой модели не равновесны, поэтому при резком изменении цвета такой способ даёт большую погрешность. Во-вторых, визуально эта метрика не всегда соответствует действительности. Например, согласно этой метрике серый цвет с коэффициентами RGB (50, 50, 50) и зелёный цвет с коэффициентами (0, 150, 0) одинаково похожи на чёрный (0, 0, 0) (см. Рис. 4, 5). Вместо среднего значения можно использовать Евклидову норму, это решит вторую проблему, но резкие изменения цвета всё равно будут приводить к ошибочным результатам.

Для хранения изображений в файлах формата JPEG разработан специальный алгоритм сжатия, допускающий малозаметные для человеческого глаза изменения цвета при сжатии. Поэтому в качестве альтернативной цветовой модели RGB в работе используется CIELAB [17].

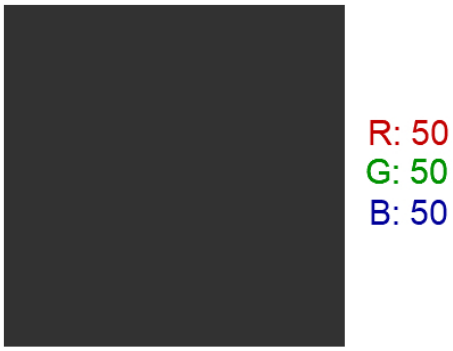

Рис. 4: Серый цвет, RGB

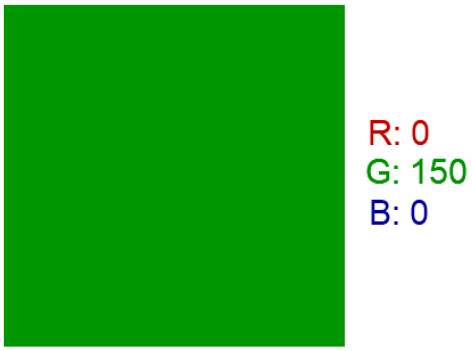

Рис. 5: Зелёный цвет, RGB

Цветовая модель	Восстановлено файлов
RGB	13 из 20
CIELAB	18 из 20

Таблица 3: Результаты использования цветовых моделей.

Эта цветовая модель специально разработана для представления цвета в максимальном приближении к человеческому восприятию. Основными компонентами этой модели являются светлота и две цветовых составляющих в диапазонах от зелёного до красного и от синего до жёлтого цветов. Для этой цветовой модели существует формула определения цветовой разницы: ΔE [13]. Она позволяет оценить степень различия цветов с точки зрения человеческого восприятия.

Чтобы определить наиболее подходящую цветовую модель для сравнения секторов, был подготовлен набор изображений формата JPEG, содержащих в себе цветовые переходы разной степени резкости. Далее в каждом из изображений секторы, содержащие часть изображения, сранивались с предыдущими при помощи Евклидовой нормы в модели RGB и с помощью формулы ΔE в модели CIELAB. Результаты сравнительного анализа двух способов сравнения секторов изображения показали, что для реализации больше подходит модель CIELAB, поскольку она позволяет с большей точностью определить подходящий сектор. Результаты сравнения приведены в Таб. 3.

Поскольку целиком хранить данные файла в оперативной памяти затратно, секторы, относящиеся к файлу, накапливаются в отдельном буфере до тех пор, пока файл не будет полностью восстановлен. Тем не менее, для обеспечения возможности возобновления ранее приостановленной задачи, в памяти хранится соответствующий экземпляр Task, содержащий минимальный набор данных для определения очередного наиболее подходящего сектора.

5. Тестирование прототипа

Для определения надёжности и качества работы прототипа проведено тестирование реализованных компонентов.

5.1. Тестирование восстановления MBR-записи

Для тестирования компонента восстановления MBR-записи были подготовлены образы жёстких дисков с разделами, отформатированными в форматах NTFS и FAT32. На двух из них было создано до четырёх разделов, а на других – четыре и больше. Такое количество разделов было выбрано для того, чтобы проконтролировать правильность восстановления MBR-записи, состоящей из одного и из нескольких секторов. Далее с каждого из них была полностью удалена MBR-запись.

Подготовленные образы были использованы в качестве входных данных для компонента восстановления MBR-записи.

В результате тестирования, на всех образах жёстких дисков MBR-запись была корректно воссоздана и доступ к разделам был восстановлен.

5.2. Тестирование восстановления данных при повреждённой MFT-таблице

Для тестирования компонента восстановления данных при повреждённой MFT-таблице был подготовлен тестовый набор фрагментированных изображений формата JPEG и проведён сравнительный анализ результатов работы.

Обзор аналогов. На сегодняшний день существует множество программ для восстановления данных с повреждённых дисков, в т.ч. программ для восстановления файлов формата JPEG, например: FTK [1], Encase [14], Scalpel [8] и т.д. Главный их недостаток заключается в том, что они не учитывают возможность фрагментации файлов на повреждённом диске. Стандартный подход в таком случае заключается в

определении заголовка файла и копировании необходимого количества памяти. Как результат, фрагментированные файлы восстанавливаются некорректно.

Существует программа Adroit Photo Forensic (APF) [2], специально предназначенная для цифрового криминалистического анализа связанного с фотографиями и изображениями. Она позволяет восстанавливать фрагментированные изображения более качественно, чем большинство аналогов. В APF реализована технология SmartCarving, которая позволяет восстанавливать фрагментированные файлы определённого формата, опираясь на их внутреннюю структуру. Эта программа будет использоваться для сравнительного анализа.

Сравнительный анализ. Для проведения сравнительного анализа было подготовлено два набора изображений в формате JPEG. В первом наборе собраны изображения, разбитые случайным образом на два фрагмента, а во втором – на четыре. Каждый набор был сохранён на индивидуальном разделе формата NTFS. В качестве входных данных для программы APF и JPEGCarver были использованы образы этих разделов с удалённой MFT-таблицей. Результаты работы APF и JPEGCarver представлены в Таб. 4.

Таблица 4: Результаты сравнительного анализа.

Количество фрагментов JPEG	JPEGCarver	Android Photo Forensic
Два фрагмента	40 из 50	39 из 50
Четыре фрагмента	34 из 50	27 из 50

На основании результатов работы APF и JPEGCarver было установлено, что наименее успешно удалось восстановить изображения с высокой степенью резкости. Чем выше резкость изображения, тем сильнее различаются детали изображения, поэтому идентифицировать нужный сектор становится сложнее.

В результате сравнительного анализа было установлено, что разра-

ботанный в рамках данной работы компонент восстановления фрагментированных JPEG-файлов восстановил больше фрагментированных файлов, чем программа APF.

Заключение

В рамках данной работы были выполнены следующие задачи:

- выполнен обзор MFT-таблицы и MBR-записи;
- исследованы способы восстановления доступа к данным при повреждении MBR-записи и MFT-таблицы; установлено, что MBR-запись можно воссоздать, определив адреса разделов и восстановив таблицу разделов, а для восстановления доступа к данным при повреждённой MFT-таблице можно воспользоваться методом сигнатурного поиска с учётом возможной фрагментации;
- определены основные компоненты прототипа; разработана архитектура прототипа;
- реализован прототип инструмента на языке C#; реализован модуль MBRRestore с поддержкой файловых систем NTFS и FAT32; реализован модуль MFTRestore с поддержкой файлов формата JPEG;
- проведено тестирование компонентов разработанного прототипа.

В дальнейшем планируется провести апробацию реализованного прототипа в инструменте криминалистического анализа Belkasoft Evidence Center [3], а также расширить функционал возможностью восстанавливать фрагментированные файлы других форматов.

Список литературы

[1] AccessData. Forensic Toolkit. — URL: https://accessdata.com/products-services/forensic-toolkit-ftk.

[2] Assembly Digital. Adroit Photo Forensics. — URL: https://www.forensicswiki.org/wiki/Adroit_Photo_Forensics.

[3] Belkasoft. Belkasoft Evidence Center. — URL: https://belkasoft.com/ec.

[4] Eugene Rodionov Alexander Matrosov David Harley. Bootkits: Past, Present Future // Virus Bulletin. — 2014.

[5] Gazet Alexandre. Comparative analysis of various ransomware virii // Journal in Computer Virology. — 2010. — Vol. 6. — P. 77–90.

[6] Heo Tejun. ATA 4 KiB sector issues. — URL: https://ata.wiki.kernel.org/index.php/ATA_4_KiB_sector_issues.

[7] Hexacorn Forensic Analysis. Sector size and MFT FILE Record size. — URL: http://www.hexacorn.com/blog/2012/05/04/sector-size-and-mft-file-record-size/.

[8] KIT Sleuth. Scalpel. — URL: https://github.com/sleuthkit/scalpel.

[9] Labs Malwarebytes. The Petya. — URL: https://blog.malwarebytes.com/threat-analysis/2016/04/petya-ransomware/.

[10] Microsoft. Attribute List Entry. — URL: https://msdn.microsoft.com/en-us/library/bb470038.aspx.

[11] Microsoft. Chkdsk. — URL: https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/chkdsk.

[12] Microsoft. File Record Segment Header. — URL: https://msdn.microsoft.com/en-us/library/bb470124.aspx.

[13] Mokrzycki W.S. Tatol M. Colour difference ΔE - A survey // Machine Graphics and Vision. — 2011. — April. — P. 383–411.

[14] OpenText. EnCase Forensic. — URL: https://www.guidancesoftware.com/encase-forensic.

[15] Ray Duncan Bill Gates. The MS-DOS Encyclopedia. — Microsoft Press.

[16] Richard Russon Yuval Fledel. NTFS Documentation. — URL: http://dubeyko.com/development/FileSystems/NTFS/ntfsdoc.pdf.

[17] Schwiegerling Jim. Field Guide to Visual and Ophthalmic Optics. — Spie Press.

[18] Wikipedia. Cylinder Head Sector // From Wikipedia, the free encyclopedia. — URL: https://en.wikipedia.org/wiki/Cylinder-head-sector.

[19] Wikipedia. GUID Partition table // From Wikipedia, the free encyclopedia. — URL: https://en.wikipedia.org/wiki/GUID_Partition_Table.

[20] Wikipedia. Logical Block Addressing // From Wikipedia, the free encyclopedia. — URL: https://en.wikipedia.org/wiki/Logical_block_addressing.

[21] Wikipedia. Master Boot Record // From Wikipedia, the free encyclopedia. — URL: https://en.wikipedia.org/wiki/Master_boot_record.

[22] de Boyne Pollard Jonathan. The gen on disc partition alignment. — URL: http://jdebp.eu./FGA/disc-partition-alignment.html.

[23] msakamoto sf. Why BIOS loads MBR into 0x7C00 in x86? // YakaBiki. — 2010. — URL: https://www.glamenv-septzen.net/en/view/6.