

**МИНОБРНАУКИ РОССИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА – Российский технологический университет»**

|  |
| --- |
| **РТУ МИРЭА** |
|  |
| **Институт кибербезопасности и цифровых технологий (ИКБ)** |
|  |
| КБ-2 «Информационно-аналитические системы кибербезопасности» |

**ОТЧЕТ О ВЫПОЛНЕНИИ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ №10**

**ПО ДИСЦИПЛИНЕ «МОДЕЛИ УПРАВЛЕНИЯ ДОСТУПОМ И ИНФОРМАЦИОННЫМИ ПОТОКАМИ В КМПЬЮТЕРНЫХ СИСТЕМАХ»**

Выполнил:

Студент 3-ого курса

Учебной группы БИСО-02-22

Зубарев В.С.

Для выполнения лабораторной работы были выбраны три фала расширения exe. Размер файлов обусловлен программой Filigrana (при попытке открыть для встраивания файл размером больше 2000кб программа переставала отвечать на действия пользователя, «зависала»)



Рисунок 1 - Образцы файлов

Программа Filigrana использует вставку изображения в пустое место файла. То есть для успешной вставки изображения необходимо иметь n пустых байт идущих подряд. В то же время n не может быть меньше 54 бит, так как заголовок bmp файла занимает именно столько места. Таким образом накладываются ограничения на exe файл и bmp вставку. В контексте лабораторной работы размер вставки не может быть меньше 54 байт, и ограничивается размером сектора диска. Причем Максимальный размер вставки рассчитывается = размер сектора – размер полезных данных на секторе.

Данные на диске объединены в строки по 16 байт, так же требуется разделение хотя бы 10-ю байтами между телом программы и встраиваемым изображением, причем разделение должно идти с двух сторон.

При попытке вставки изображения в первый файл(167 кб) программ нашла самую длинную последовательность нулевых битов.

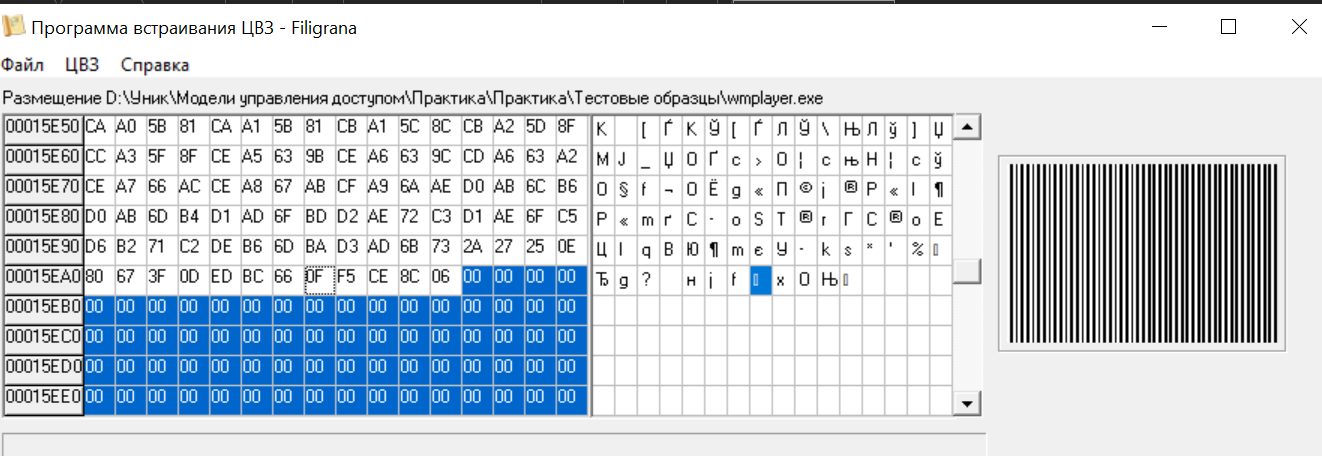


Рисунок 2- Начало вставки

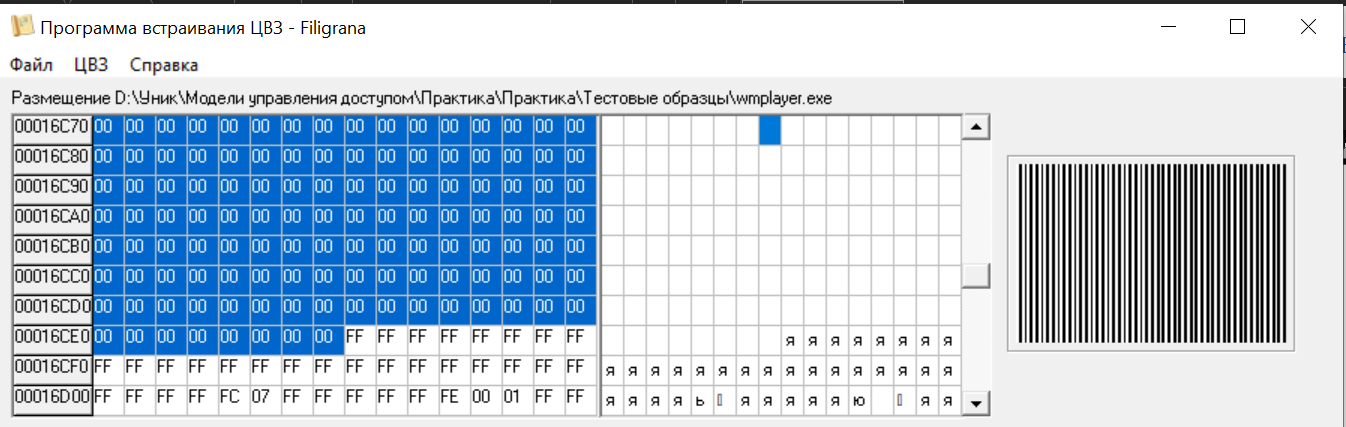


Рисунок 3- Конец вставки

Для вставки изображения доступны строки с 15EB0 до 16СD0. То есть всего 3616 байт

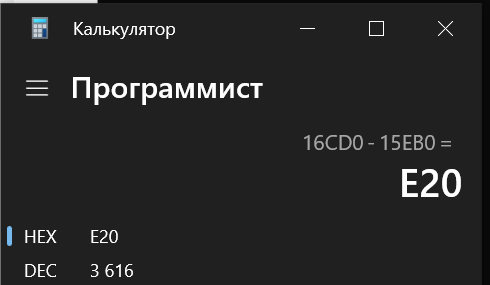


Рисунок 4- Вычисление максимального объема ЦВЗ

Из 3616 байт по 10 необходимо оставить на разграничение от начала и конца встраиваемого файла, получаем максимальный возможный размер 3596 байт. Об этом сообщает и программа при попытке встроить файл, размер которого больше указанного.

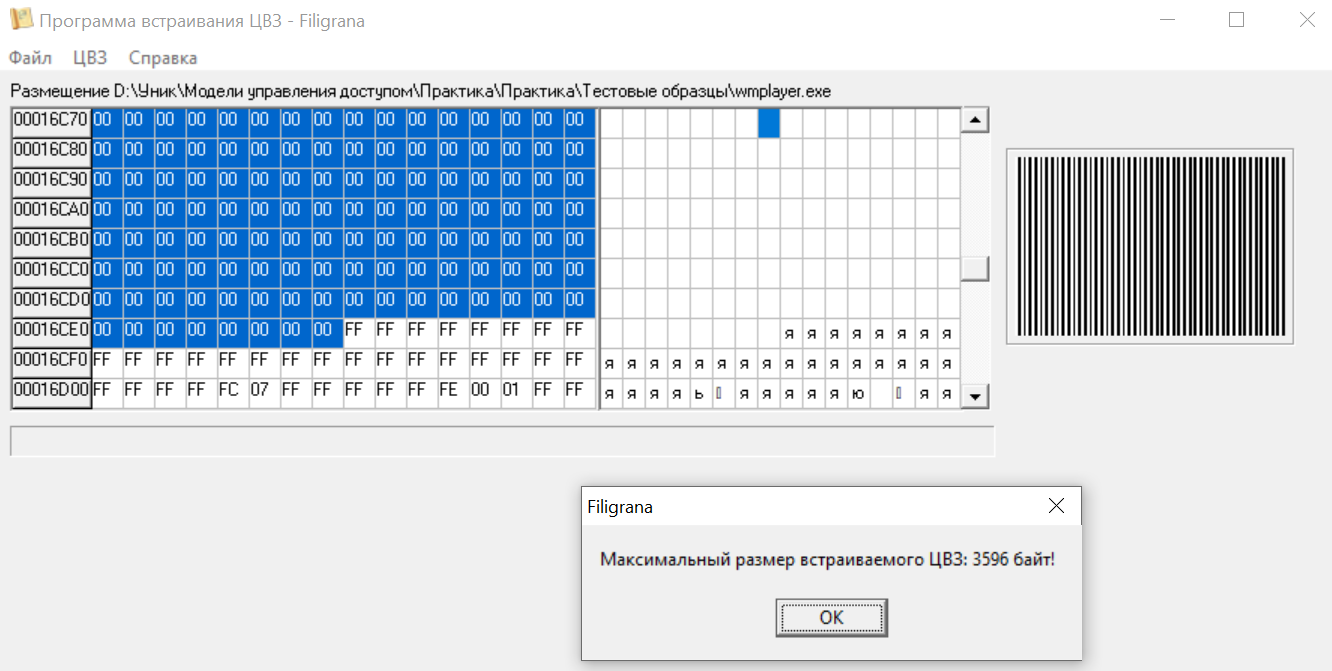


Рисунок 5 – Демонстрация ограничений для встраиваемого файла

Аналогично были получены максимальный размеры файлов для каждого из трех exe файлов (445,3596, 31901 байт). Для генерации изображений с точностью до байт был написан скрип на языке Python.

Листинг программы

import struct  
def create\_bmp(file\_size):  
 # Размеры заголовков  
 header\_size = 54  
 # Количество байт для данных пикселей  
 pixel\_data\_size = file\_size - header\_size  
 # Количество пикселей (по 3 байта на пиксель)  
 pixel\_count = pixel\_data\_size // 3  
 # Размеры изображения (пока что будем фиксировать их как квадратное изображение)  
 width = int(pixel\_count \*\* 0.5)  
 height = width  
 # Паддинг, чтобы строка данных была кратна 4 байтам  
 row\_size = (width \* 3 + 3) & (~3)  
 data\_size = row\_size \* height  
 # Заголовок файла (14 байт)  
 file\_header = b'BM'  
 file\_header += struct.pack('<I', header\_size + data\_size) # общий размер файла  
 file\_header += struct.pack('<HH', 0, 0) # зарезервировано  
 file\_header += struct.pack('<I', header\_size) # смещение до данных изображения  
 # Заголовок информации (DIB Header, 40 байт)  
 dib\_header = struct.pack('<I', 40) # размер DIB заголовка  
 dib\_header += struct.pack('<I', width) # ширина изображения  
 dib\_header += struct.pack('<I', height) # высота изображения  
 dib\_header += struct.pack('<H', 1) # количество цветовых плоскостей  
 dib\_header += struct.pack('<H', 24) # количество бит на пиксель (24 бита = RGB)  
 dib\_header += struct.pack('<I', 0) # тип сжатия (0 = без сжатия)  
 dib\_header += struct.pack('<I', data\_size) # размер данных изображения  
 dib\_header += struct.pack('<I', 0) # горизонтальное разрешение  
 dib\_header += struct.pack('<I', 0) # вертикальное разрешение  
 dib\_header += struct.pack('<I', 0) # количество цветов в палитре  
 dib\_header += struct.pack('<I', 0) # важные цвета (0 - не важно)  
 # Данные изображения (pixel data)  
 pixel\_data = b''  
 for i in range(pixel\_count):  
 # Чередующиеся чёрно-белые пиксели  
 color = (255, 255, 255) if i % 2 == 0 else (0, 0, 0) # чёрный и белый пиксели  
 pixel\_data += struct.pack('<BBB', \*color)  
 # Паддинг до 4 байтов в каждой строке  
 pixel\_data\_padded = b''  
 for y in range(height):  
 row = pixel\_data[y \* width \* 3: (y + 1) \* width \* 3]  
 row += b'\x00' \* (row\_size - len(row)) # добавление padding  
 pixel\_data\_padded += row  
 # Составляем итоговый файл  
 with open('image30000.bmp', 'wb') as f:  
 f.write(file\_header)  
 f.write(dib\_header)  
 f.write(pixel\_data\_padded)  
  
# Генерация BMP-файла  
file\_size = 30000 # Размер файла в байтах (включая заголовки)  
create\_bmp(file\_size)

Результатом работы были три тестовых изображения формата bmp.

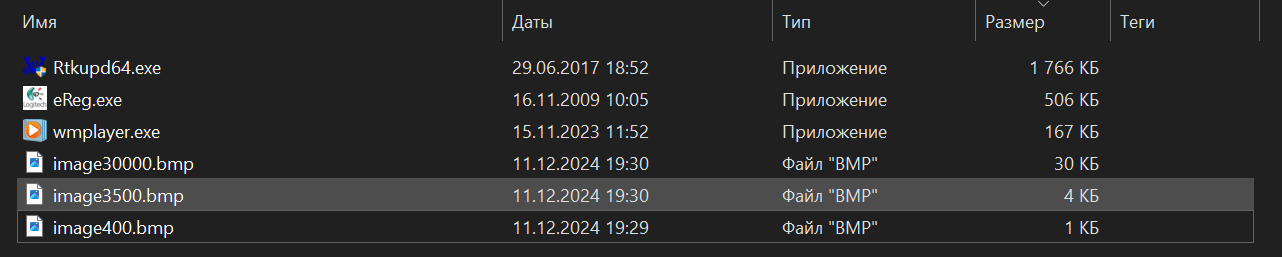


Рисунок 6 – Сгенерированные изображения

Внедрение картинки происходит в 4 этапа:

1. Чтение картинки, представление структуры BMP файла в бинарном виде. Файл не может быть меньше 57 байта (54 для заголовка + 3 байта для одного пикселя, так как bmp использует кодировку 24 битами на пиксель). Максимальный возможный размер зависит от файловой системы для 32 битных 232 байт для 64-битных 263 байт.
2. Чтение exe файла, представление его в бинарном виде. Файл не может быть меньше 512 байт. Максимальный возможный размер зависит от файловой системы для 32 битных 232 байт для 64-битных 263 байт.
3. Поиск места. На данном этапе сканируется exe файл на наличие не используемых(пустых) битов. Максимальный и минимальный размер exe файла отсеется неизменным, так же, как и минимальный размер файла bmp, однако максимальный размер обусловлен длинной непрерывной цепочки пустых байтов в структуре exe файла.
4. Упаковка. На этом этапе, в случае если размер bmp удовлетворяет условиям вставки, происходит запись структуры bmp файла на место пустых битов exe файла.

Для каждого из трех файлов были проведены попытки упаковки изображения.

Файл 1 (мастер установки Windows media player, 167кб)

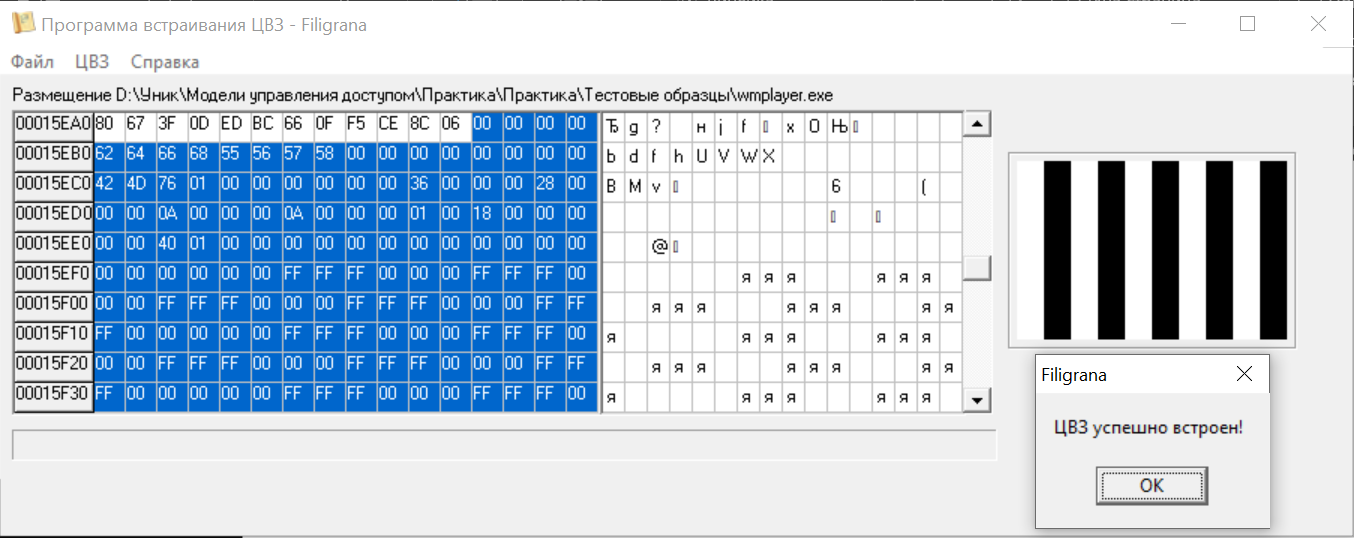


Рисунок 7 – Вставка 1 (imgae400.bmp, 400 байт)

Вставка ЦВЗ начинается с 7 байта строки 15EFO

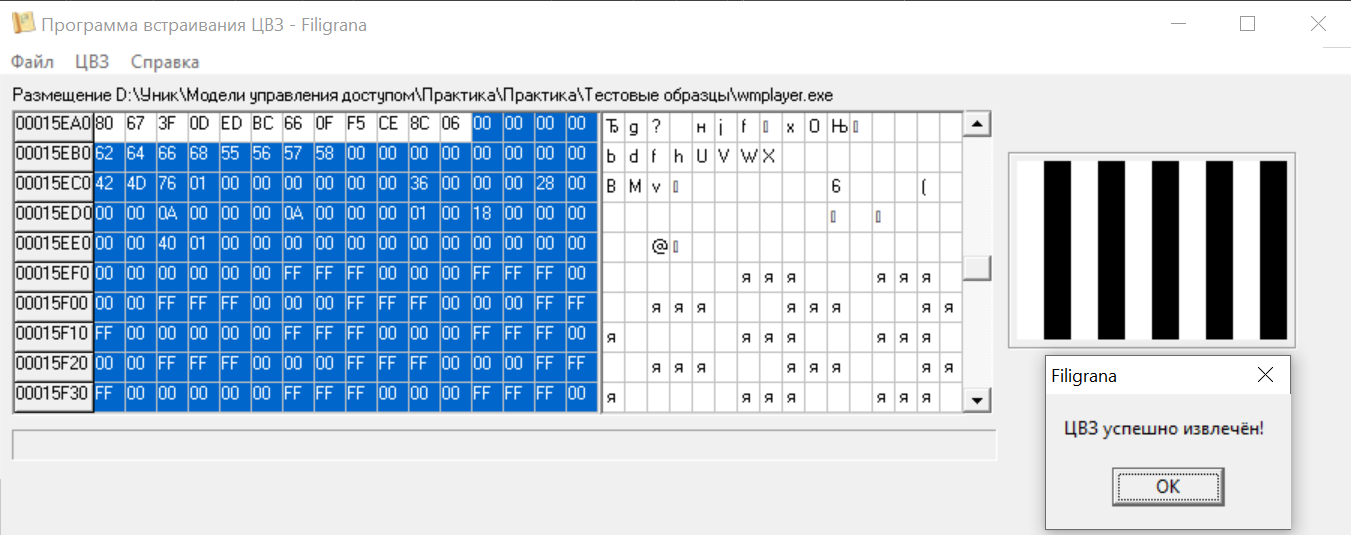


Рисунок 8 – Извлечение 1 (imgae400.bmp, 400 байт)

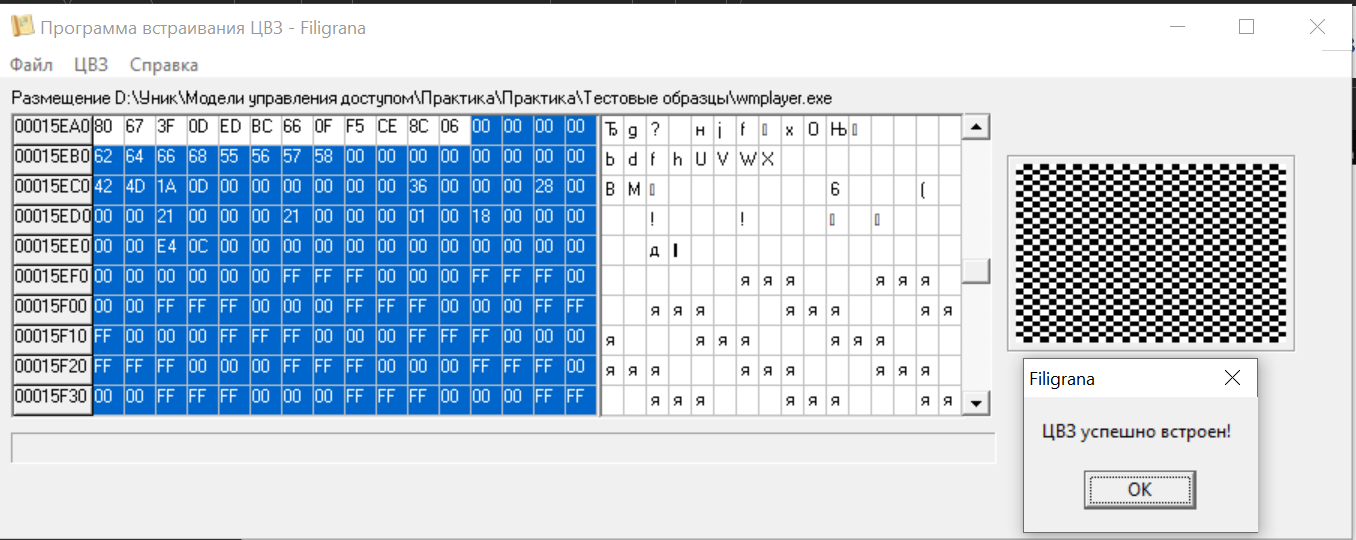


Рисунок 9 – Вставка 2 (imgae3500.bmp, 3500 байт)

Вставка ЦВЗ начинается с 7 байта строки 15EFO

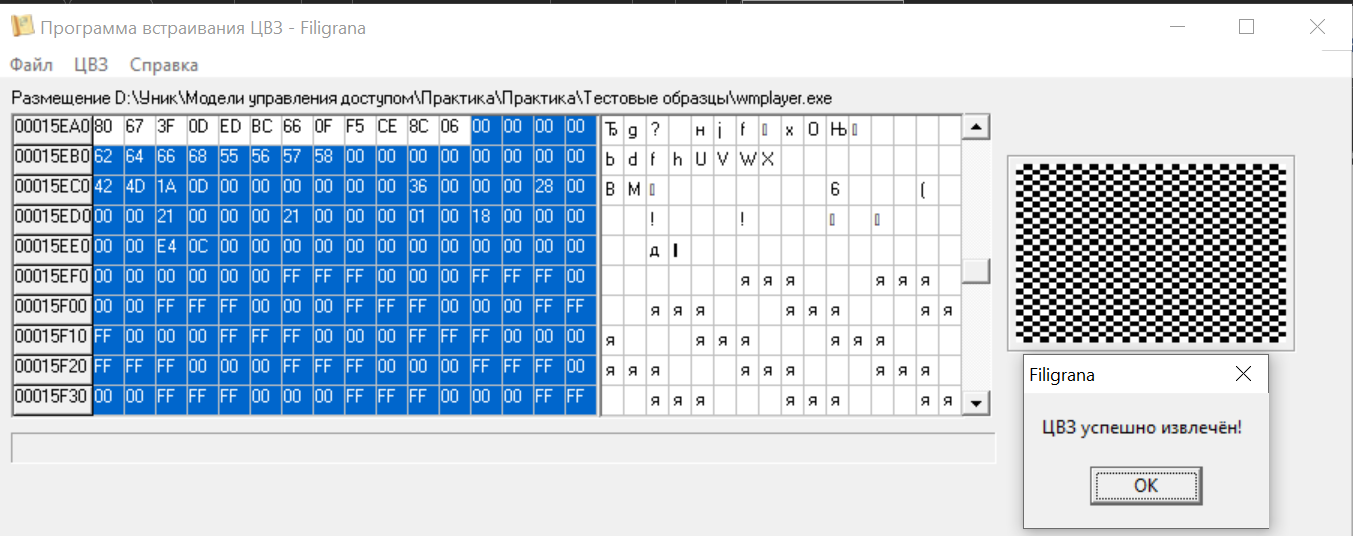


Рисунок 10 – Извлечение 2 (imgae3500.bmp, 3500 байт)

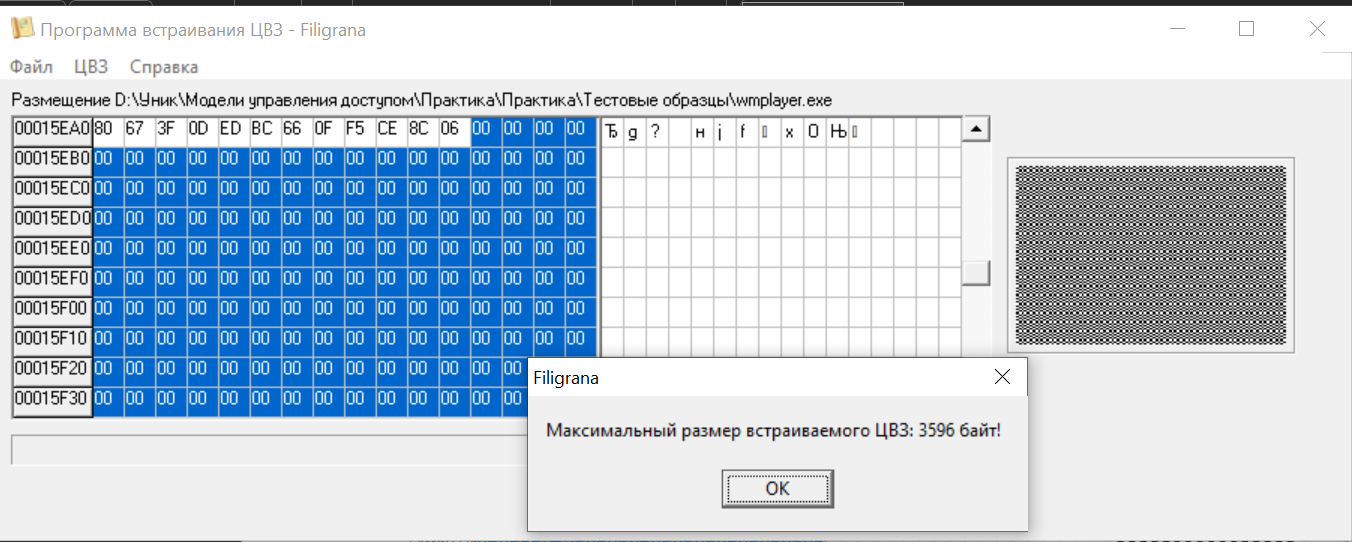


Рисунок 11 – Вставка 3 (imgae30000.bmp, 30000 байт)

Изображение превышает размер найденного контейнера для вставки. Так как изображение не было вставлено, то оно не может быть извлечено.

Файл 2 (Мастер установки Logitech webcam software, 506 кб)

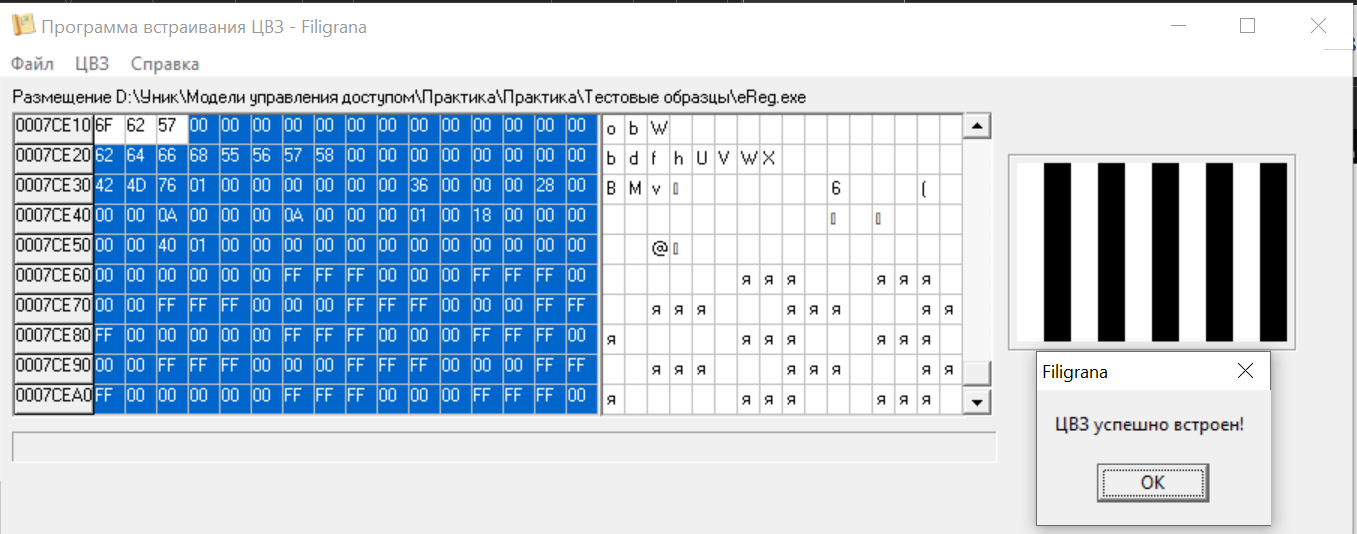


Рисунок 12 – Вставка 1 (imgae400.bmp, 400 байт)

Вставка ЦВЗ начинается с 7 байта строки 7CE60

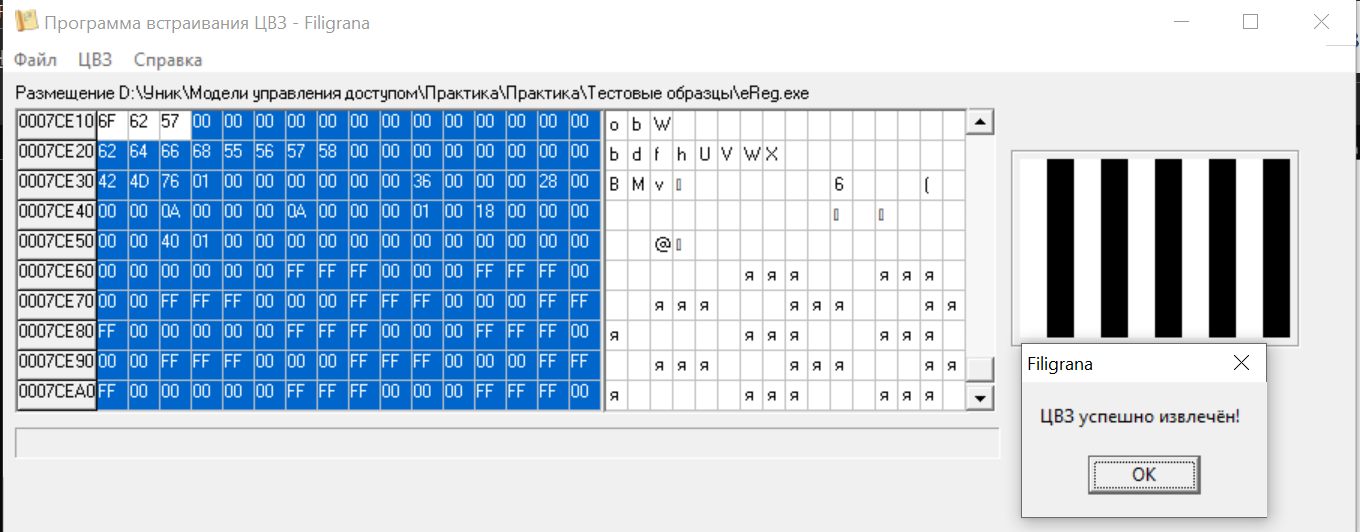


Рисунок 13 – Извлечение 1 (imgae400.bmp, 400 байт)

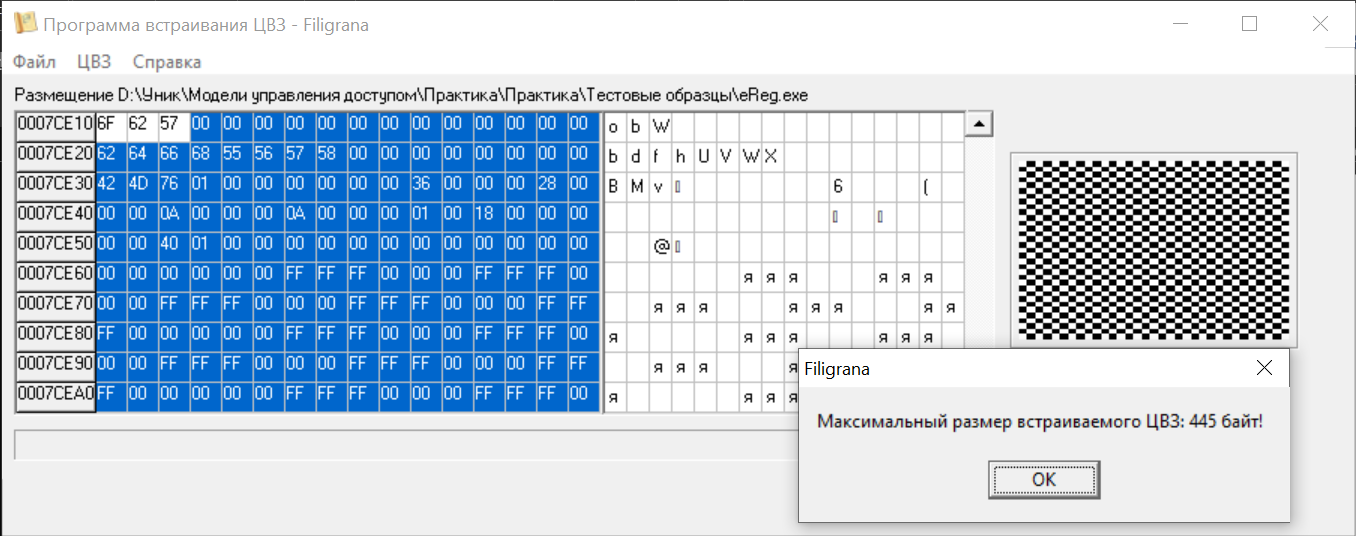


Рисунок 14 – Вставка 2 (imgae3500.bmp, 3500 байт)

Изображение превышает размер найденного контейнера для вставки. Так как изображение не было вставлено, то оно не может быть извлечено.

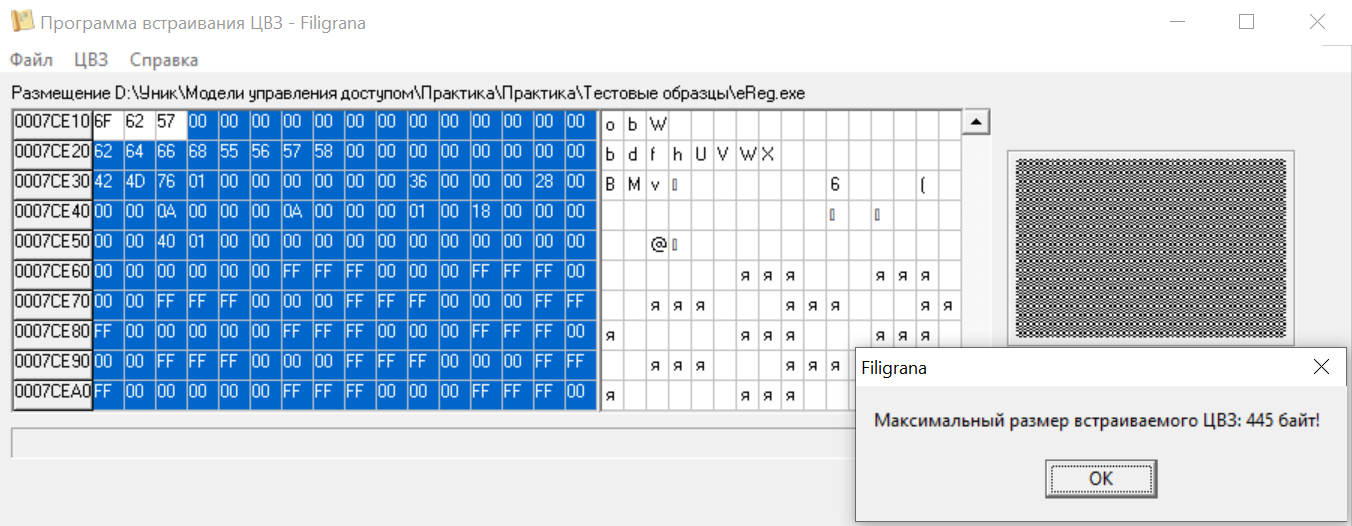


Рисунок 15 – Вставка 3 (imgae30000.bmp, 30000 байт)

Изображение превышает размер найденного контейнера для вставки. Так как изображение не было вставлено, то оно не может быть извлечено.

Файл 3 (мастер установки Realtek HD Audio, 1766кб)

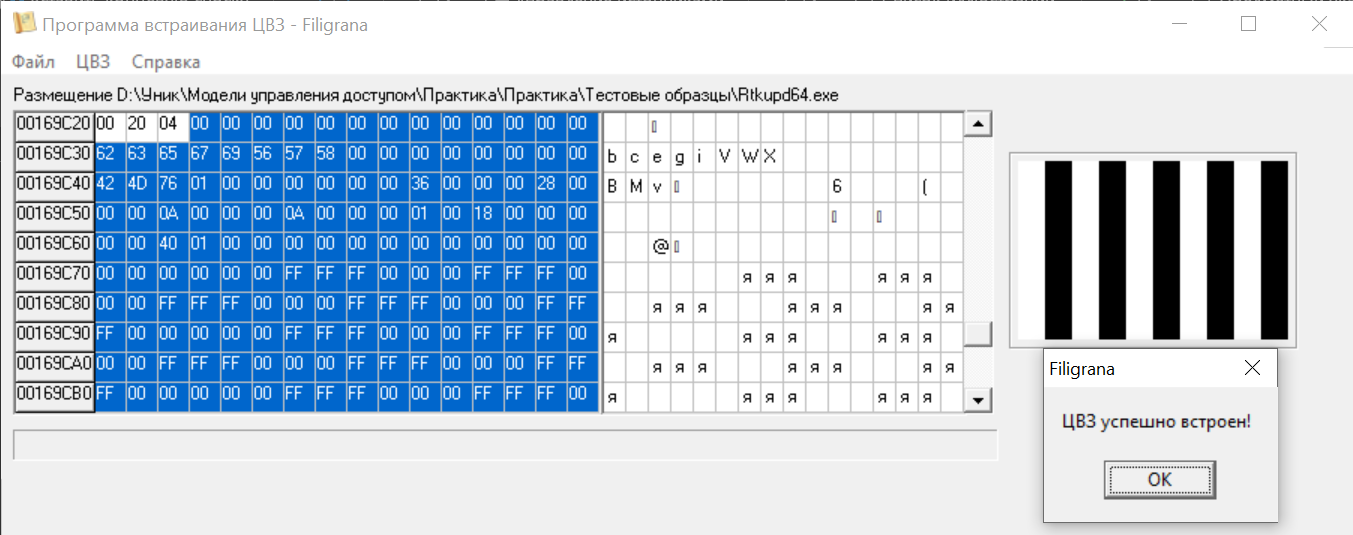


Рисунок 16 – Вставка 1 (imgae400.bmp, 400 байт)

Вставка ЦВЗ начинается с 7 байта строки 169C70

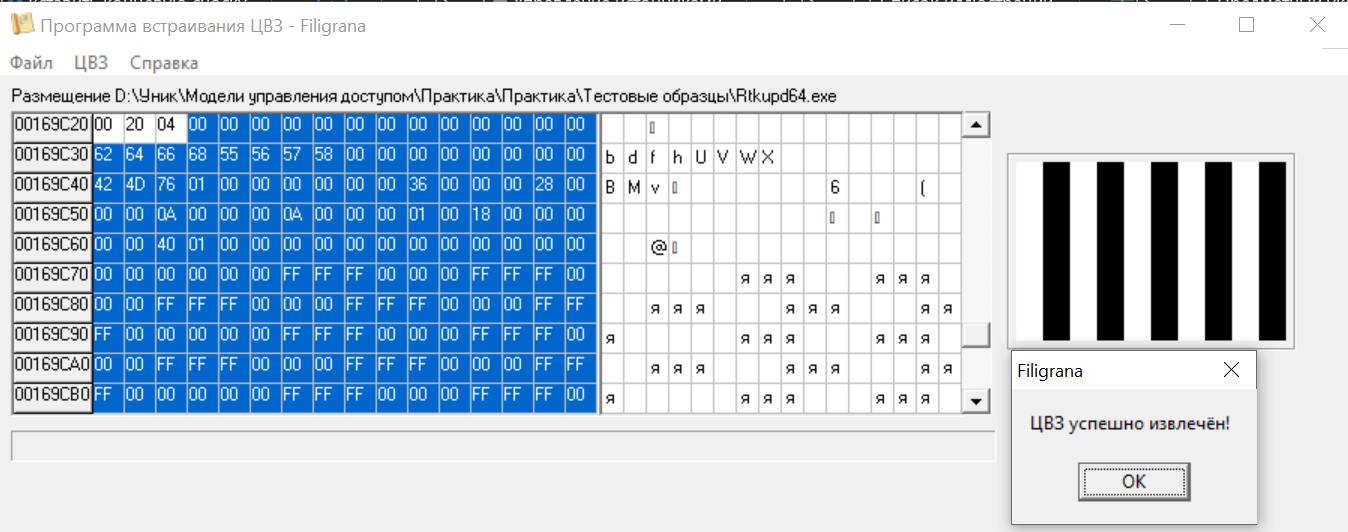


Рисунок 17 – Извлечение 1 (imgae400.bmp, 400 байт)

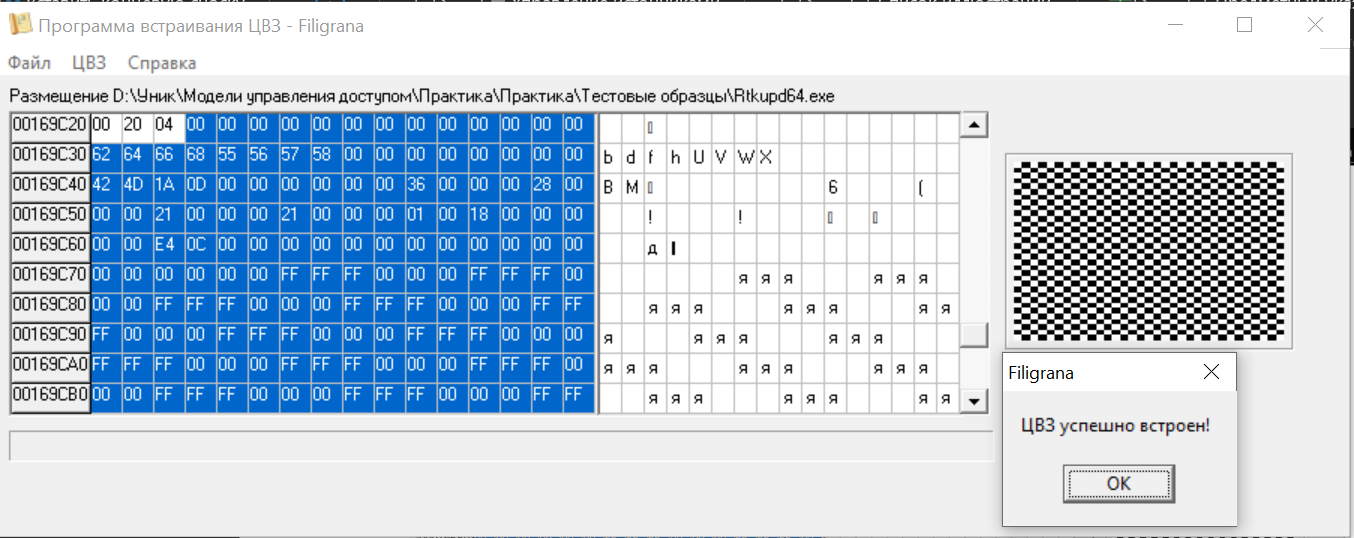


Рисунок 18 – Вставка 2 (imgae3500.bmp, 3500 байт)

Вставка ЦВЗ начинается с 7 байта строки 169C70

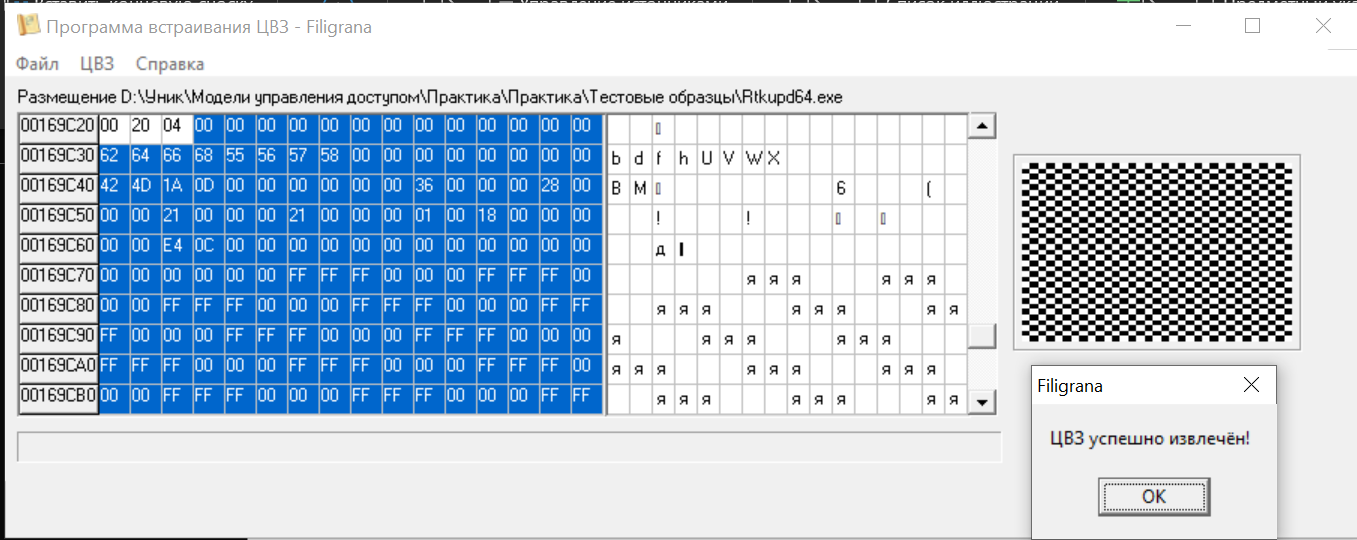


Рисунок 19 – Извлечение 2 (imgae3500.bmp, 3500 байт)

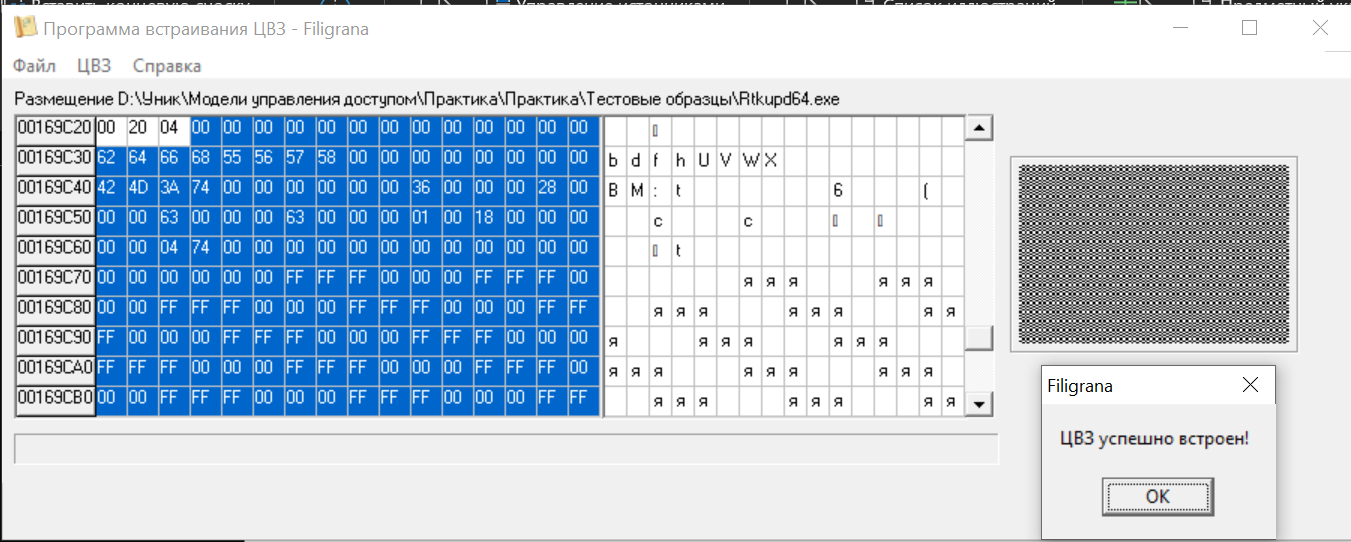


Рисунок 20 – Вставка 3 (imgae30000.bmp, 30000 байт)

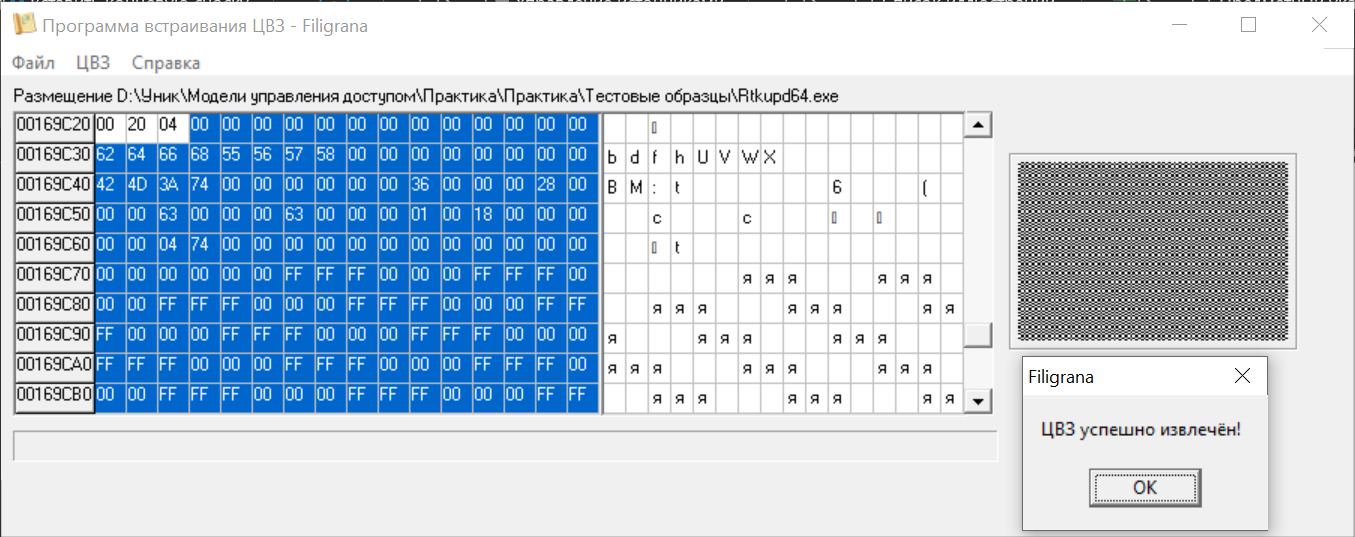


Рисунок 21 – Извлечение 3 (imgae30000.bmp, 30000 байт)

2 5 6