**Практика 4.**

**Обратная разработка. Базовый анализ исполняемых файлов.**

**Целью** данной практики является получение базовых навыков анализа исполняемых файлов и использования инструментов для обратной разработки.

Для выполнения данной практики можно использовать различные инструменты, однако все задания можно выполнить с помощью IDA Pro.

**1. Первоначальные действия.**

 При анализе любых исполняемых файлов чаще всего в начале стоит ряд задач, решаемых довольно быстро. Например, определение микропроцессорной архитектуры команд или разрядности файла. Чаще всего после решения данных задач и загрузки исполняемого файла в IDA Pro, производится первичный анализ файла. В первую очередь просматриваются строки, имеющиеся в исполняемом файле. Строки могут помочь понять некоторую логику работы приложения, так-как можно проследить, где какая строка используется, в связи с этим понять назначение того или иного блока. Список строк в IDA Pro можно получить с помощью нажатия “Shift+f12”.

 В открывшемся окне будут отображены все автоматически распознанные строки, для того, чтобы попасть на само место хранения строки в исполняемом файле достаточно 2 раза нажать ЛКМ по интересующей строке. После, для того чтобы узнать, где данная строка используется, можно получить все перекрёстные ссылки на неё путём нажатия горячей клавиши “x”, при выделенной метке, отвечающей за строку (пример на рисунке ниже).



Необходимо навести на метку (имя) для данной строки (обычно располагается слева от самого значение) и нажать горячую клавишу “x”, после откроется окно (как показано на рисунке), где будут отображены все ссылки (то есть все использования данной строки, если они имеются) на данную строку в программе. Исходя из этих данных, можно определять предназначение тех или иных блоков (например, блок, отвечающий за верно введённый ключ).

 **2. Представление различных типов данных.**

 IDA Pro умеет автоматически определять различные типы данных, однако зачастую она делает это не совсем верно, и требуется «помогать» ей в этом. Для этого существуют функции преобразования данных. Например, для преобразования какого-либо значения в ASCII символ можно использовать клавишу “R” (пример на рисунке ниже).



После нажатия на “R”, выделенное нами значение (там, где находится курсор) преобразуется в ASCII символ ‘8’, соответственно таким же образом можно преобразовать все последующие значения.

**3. Анализ простых условий.**

 Практически все более-менее большие программы имеют многочисленные ветвления. Это связано с наличием различны сравнений, проверки условий и прочего. Для анализа простых условий можно использовать графическое представление, для более удобного отображения ветвления (рисунок ниже).



Можно заметить мнемонику “cmp” перед мнемоникой “jnz”, которая является условным переходом в один из блоков, если переход не произойдёт, продолжится выполнение следующей команды, которая находится в соседнем блоке. Как можно заметить, необходимо чтобы в регистре eax, к моменту сравнения находилось значение 297h (h-указание на то, что величина является шестнадцатеричной). Для решения данной задачи необходимо проследить, какие изменения происходят с данным регистром до этого момента и попытаться выяснить, от каких данных и действий зависит конечное значение. Для этого необходимо проанализировать небольшой участок кода, представленный на рисунке ниже.



Как можно заметить, в регистр eax помещается значение локальной переменной, которая инициализируется путём ввода с клавиатуры. То есть изначально в регистре eax будет находится введённое пользователем значение. Далее это значение проходит через ряд простым математических операций, таких как сложение, побитовый сдвиг вправо, сложение по модулю два с числом, и сложение по модулю два с другой локальной переменной, ранее проинициализированной. Необходимо провести данные операции в обратную сторону с финальным числом (то есть с числом, с которым производится сравнение) и найти число, которое необходимо ввести с клавиатуры для получения верного выполнения финальной проверки.

**Задание на практику:**

Произвести обратную разработку 3 исполняемых файлов, в первом и втором исполняемых файлах найти контрольные значения. В третьем исполняемом файле найти число (или числа, если их несколько), при вводе которого (которых) происходит выдача сообщения о верности секретного ключа.

Решения всех заданий должны быть подробно описаны и содержать скриншоты. Для последнего задания должна быть приведена формула, по которой происходит преобразование введённого значения, и обоснована верность полученного ответа.