Задание было сделано совместо с Максимом Хоронеко.

Сначала мы испльзовали ILSpy, чтобы достать IL. Обнаружили, что код обфусцирован. Прогнали ассембли через de4dot, что несколько помогло. Получили более-менее читаемые сорцы.

Потом обнаружили, что не дизассемблировались сгенерированные компилятором типы и методы. Включили эту опцию в декомпиляторе.

Все равно проследить, что происходит, сразу не получилось. Поэтому мы использовали ildasm, чтобы получить pdb, скомпилировали заново и начали дебажить по IL-коду.

Наконец нашли, где началась проверка ключа, начали очень тщательно отслеживать, что происходит в этом месте в декомпилированных сорцах. Начали удалять «мусорные» развилки, мусорные классы, мусорные переменные, добавленные обфускатором. Наконец получили код, которые генерирует некий массив целых чисел.

Некоторое время провозились, прежде чем понять, что все, что оставшийся код делает, - это проверка, что введенный пользователем код равен сгенерированному...

В итоге генерация кода сводится к этому:

private static string GetSerial()

{

var networkInterface = NetworkInterface.GetAllNetworkInterfaces().First();

var addressBytes = networkInterface.GetPhysicalAddress().GetAddressBytes();

var dateBytes = BitConverter.GetBytes(DateTime.Now.Date.ToBinary());

var numArray = addressBytes

.Select((b, index) => b ^ dateBytes[index])

.Select(x => x < 999 ? x \* 10 : x)

.ToArray();

var result = string.Join("-", numArray.Select(i => i.ToString()));

return result;

}

Если бы это был настоящий код, я полагаю, что был бы использован настоящий алгоритм шифрования (какая-нибудь криптостойкая хеш-функция), а не простой XOR данных, и не стал бы полагаться на обфускацию. При большом желании понять, что делает обфусцированный код, не так уж сложно.