1. **ЗАЩИТА ВНОВЬ СОЗДАННЫХ ПРОГРАММ**

Нередко, обычные пользователи невнимательно относятся к вопросам защиты информации и компьютерной безопасности. Для примера можно привести отсутствие в операционной системе антивирусного сканнера, отключенный либо отсутствующий фаервол и т.д. И как результат, на этом компьютере, скорее всего, будут десятки, если даже не сотни заражённых различными вирусами объектов. К конечном счёте, это ведёт к краху операционной системы, последующему форматированию системного диска и установке новой операционной системы. Но, если на остальных дисках компьютера остался хотя бы один заражённый файл, достаточно один раз запустить его и ситуация с заражением повторится снова.

Во избежание этой ситуации, в код создаваемой программы можно внедрить алгоритм проверки исполняемого файла на целостность. Этот метод никаким образом не мешает заражению исполняемого файла, или выполнению вредоносного кода, и носит сугубо информативный характер. Но это поможет вовремя предупредить пользователя о заражении и предпринять соответствующие действия. Сам алгоритм проверки целостности заключается в высчитывании *контрольной суммы* файла - некоторого значения, рассчитанного по набору данных путём, применения определённого алгоритма. Несмотря на своё название, контрольная сумма не обязательно вычисляется путем суммирования данных. Существует множество технологий подсчёта контрольных сумм и все они используют разнообразные методы расчёта: обычное сложение, вычитание, сложение по модулю и т.д. Но в результате вычисления появляется значение, уникальное для каждого файла. А саму проверку можно организовать путём сравнения текущей контрольной суммы с исходным значением, записанным в программе. Ведь, при заражении вирусом контрольная сумма файла неизбежно изменяется. Такого рода проверку полезно добавлять и во избежание потерь данных при обмене файлами в сети интернет, или даже как защиту от начинающих взломщиков программ.

Так же следует опасаться вирусов, поражающих исходный код программы. Вирусы данного типа поражают или исходный код программы, или её компоненты (OBJ-, LIB-, DCU-файлы), а так же VCL и ActiveX компоненты. После компиляции программы вредоносный код оказываются в неё встроенным. В настоящее время широкого распространения данные вирусы не получили. Хотя, не так уж и давно, в августе 2009 года, "Лаборатория Касперского" обнаружила вирус *Virus.Win32.Induc.a,* заражающий Delphi-приложения на этапе разработки. Данный вирус внедряется в исходный файл базовых констант Delphi и изменяет его, в результате чего получается модифицированный файл базовых констант. В дальнейшем все программы, создаваемые в заражённой среде, содержат код нового вируса. Вредоносный код не несет ни какой функциональной нагрузки, помимо самого заражения, скорее он предназначен для демонстрации знаний и умений какого-нибудь начинающего хакера.

Хотя вирусы, без сомнения, представляют большую опасность, все они выполняются по стандартному алгоритму, использованному вирусописателем. Большинство вредоносных программ ориентировано на заражение как можно большего количества компьютеров, не останавливаясь на каком-то отдельном приложении.

И если рассматривать защиту какого-то отдельного приложения - главную опасность для него представляют не вирусы. Если созданная программа оказалась полезной и завоевала популярность у пользователей, автор решает получить прибыль с данного продукта. И он, к примеру, внедряет процедуру регистрации пользователей программы, ограничивая её функциональность в бесплатном режиме. Чем и привлекает внимание злоумышленников, предпочитающих пользоваться полнофункциональной версией программы бесплатно. Современный хакер имеет в своем арсенале набор разнообразных утилит для взлома. Их можно подразделить на несколько категорий:

* Отладчики. Позволяют прерывать выполнение программы при достижении заранее заданных условий, производить пошаговое выполнение программы, изменять содержимое памяти и регистров и т.п. Наиболее популярным, удобным и мощным является отладчик SoftICE, который при достаточно примитивном интерфейсе обладает приличными возможностями и весьма стабильно работает.
* Дизассемблеры. Производят дизассемблирование программы для дальнейшего изучения полученного кода. Один из наиболее мощных и популярных – IDA.
* Средства мониторинга – это набор утилит, отслеживающих операции с файлами, реестром, портами и сетью. Наиболее популярными являются Filemon и Regmon.
* Средства пассивного анализа программы. Показывают разную информацию о программе - извлекают ресурсы, показывают связи, используемые библиотеки. Классический пример - утилита *DEPENDS.EXE* из комплекта Visual Studio. Она показывает, какие библиотеки используются программой, и какие функции импортируются.
* Прочие утилиты. Их великое множество - это разнообразные редакторы, анализаторы ...

Программу, которая, кроме проверки кода регистрации, не имеет никакой защиты, взломает за пару минут даже начинающий хакер. Для того, чтобы не облегчать жизнь взломщикам, существует несколько методов защиты данных:

* Упомянутый выше метод *проверки целостности файла*;
* Метод замены используемых в программе функций API на аналогичные алгоритмы, написанные на Ассемблере. Все стандартные функции Windows легко расшифровываются дизассемблером. Тоже относится и к библиотекам VCL, используемых в большом количестве компиляторов. Так же не стоит использовать осмысленные имена функций защиты, причём даже пользовательских.
* Метод использования алгоритмов шифрования;
* Метод использования упаковщиков исполняемых файлов;

И даже эти методы не смогут остановить опытного хакера. Они лишь усложнят взлом программы на некоторое время. Программы, которую невозможно взломать, не существует. Всё зависит от времени и сложностей, с которыми столкнётся взломщик.