ЭЛЕКТРОННЫЕ КЛЮЧИ ЗАЩИТЫ ПО

Демидов С.А., 4 курс,
Слабко Ю.И., преподаватель,
УО «Белорусский национальный технический университет»

С развитием рынка информационных и коммуникационных технологий актуальной стала проблема несанкционированного распространения программных продуктов (попросту, «пиратство»).

Электронный ключ (dongle) – это небольшое устройство, предназначенное для защиты программного обеспечения от несанкционированного копирования. Основная функциональность состоит в хранении небольшого количества данных и преобразования данных по встроенному алгоритму.

Один ключ можно использовать для защиты нескольких приложений. При этом каждое защищаемое приложение может использовать различные данные, хранящиеся в памяти ключа и различные аппаратные алгоритмы. Под аппаратными алгоритмами в данном случае понимаются алгоритмы, реализующие специальные функции вида Y=F(X), способные преобразовывать информацию таким образом, что по значению Y невозможно определить аргумент X, то есть функции не имеют обратных.

Практически все существовавшие ранее электронные ключи предоставляли лишь возможность для проверки своего наличия, шифрования и расшифровки данных универсальным для всех ключей алгоритмом или для хранения данных. В общих чертах схема работы с электронными ключами описывалась так:

– приложение «привязывается» к ключу при помощи специального ПО;
– во время работы защищенное приложение обменивается с ключом информацией, с помощью которой ключ «опознается»;
– если ключ отсутствует или имеет неверные параметры, то приложение не работает.

Таким образом, защищенную программу нет смысла копировать, т. к. без ключа она будет неработоспособна. Однако такая схема позволет как просто отключать проверку ключа в программе, так и создать «универсальные» программные эмуляторы, которые создавали видимость полной функциональности ключа для программного обеспечения. При использовании современных электронных ключей принцип защиты кардинально изменяется.

После вынесения функций из программы в ключ, подобные технологии взлома становятся невозможными, так как для этого взломщику необходимо знать программный код, который был помещен в ключ и является абсолютно уникальным для каждого разработчика. Исполняемый в ключе код не может быть извлечен для анализа и последующей эмуляции благодаря используемому высокозащищенному микропроцессору, выполненному по технологии смарткарт.

Есть **два способа защиты** с использованием электронных ключей – *автоматическая* (или навесная) защита и защита *при помощи функций API*. В первом случае защищается исполняемый программный модуль, во втором – функции защиты встраиваются в исходный код программы.

ПолесГУ

Автоматическая защита применяется к уже готовым программам, позволяет построить довольно мощную защиту буквально за несколько щелчков мышью. Автоматическая защита устанавливается при помощи соответствующей утилиты (NSDKEY.EXE – для DOS-приложений, NWKEY.EXE – для 16-битных Windows-приложений и NWKEY32.EXE – для 32-битных Windows-приложений). Однако модуль автозащиты, внедряемый в программу, не может составлять с ней единого целого и есть опасность, что хакер найдёт возможность разделить модуль автозащиты и приложение. Автоматическая защита обладает рядом полезных свойств: защита от отладчиков, кодирование тела защищенного приложения и программных оверлеев, использование аппаратного алгоритма.

Функции API предназначены для выполнения различных операций с ключом: поиск нужного ключа, чтение/запись его памяти, запуск аппаратных алгоритмов ключа и преобразование данных приложения с их помощью. Суть метода состоит в том, что разработчик проектирует систему защиты сам, используя функции API. Затем, система защиты встраивается в приложение на уровне исходных текстов, в результате чего она составляет единое целое с защищаемым приложением. При API-защите рекомендуется использовать несколько функций. Их вызовы необходимо распределить по коду приложения и перемешать переменные функций с переменными приложения. Таким образом, защита API оказывается глубоко внедренной в программу. Обязательным является использование алгоритмов преобразования данных. Кодирование информации делает бессмысленным удаление вызовов функций API, ведь при этом данные не будут декодированы. Эффективный прием усложнения логики защиты – откладывание реакции программы на коды возврата функций API. В этом случае программа принимает решение о дальнейшей работе спустя какое-то время после получения кодов возврата. Что заставляет взломщика прослеживать сложные причинно-следственные связи и исследовать в отладчике слишком большие участки кода. API-защита существует для наиболее распространенных языков и сред программирования. В их числе различные версии C/C++, Pascal, Delphi, Visual Basic, Java, Fortran и др.

ПолесГУ