Основные подходы к защите от несанкционированного копирования

Функции средств защиты от копирования

Инсталлированная программа для защиты от копирования при каждом запуске должна выполнять следующие действия:

- анализ аппаратно-программной среды компьютера, на котором она запущена, формирование на основе этого анализа текущих характеристик своей среды выполнения
- проверка подлинности среды выполнения путем сравнения ее текущих характеристик с эталонными, хранящимися на винчестере
- блокирование дальнейшей работы программы при несовпадении текущих характеристик с эталонными

Варианты выполнения процесса проверки подлинности:

- с использованием множества операторов сравнения того, что есть, с тем, что должно быть
- с использованием механизма генерации исполняемых команд в зависимости от результатов работы защитного механизма
- с использованием арифметических операций.

Методы снятия защиты от копирования

- Статические методы предусматривают анализ текстов защищенных программ в естественном или преобразованном виде.
- Динамические методы предусматривают слежение за выполнением программы с помощью специальных средств снятия защиты от копирования.

Основные методы защиты от копирования

- Криптографические методы
- Метод привязки к идентификатору
- Методы, основанные на работе с переходами и стеком
- Манипуляции с кодом программы
- Методы противодействия динамическим способам снятия защиты программ от копирования

Криптографические методы

Для защиты инсталлируемой программы от копирования при помощи криптографических методов инсталлятор программы должен выполнить следующие функции:

- анализ аппаратно-программной среды компьютера, на котором должна будет выполняться инсталлируемая программа, и формирование на основе этого анализа эталонных характеристик среды выполнения программы
- запись криптографически преобразованных эталонных характеристик аппаратно-программной среды компьютера на жесткий диск

Можно выделить два основных метода защиты от копирования с использованием криптографических приемов:

- с использованием односторонней функции;
- с использованием шифрования (которое также может использовать односторонние функции).

Односторонние функции это функции, для которых при любом x из области определения легко вычислить f(x), однако почти для всех y из ее области значений, найти y=f(x) вычислительно трудно.

Метод привязки к идентификатору

Заключается в том, что на жестком диске при инсталляции защищаемой от копирования программы формируется уникальный идентификатор, наличие которого затем проверяется инсталлированной программой при каждом ее запуске. При отсутствии или несовпадении этого идентификатора программа блокирует свое дальнейшее выполнение.

Методы, основанные на работе с переходами и стеком

Данные методы основаны на включение в тело программы переходов по динамически изменяемым адресам и прерываниям, а также самогенерирующихся команд (например, команд, полученных с помощью сложения и вычитания). Вместо команды безусловного перехода может использоваться возврат из подпрограммы. Предварительно в стек записывается адрес перехода, который в процессе работы программы модифицируется непосредственно в стеке.

Стек определяется непосредственно в области исполняемых команд, что приводит к затиранию при работе со стеком. Этот способ применяется, когда не требуется повторное исполнение кода программы.

Манипуляции с кодом программы

1. включение в тело программы «пустых» модулей (заключается во включении в тело программы модулей, на которые имитируется передача управления, но реально никогда не осуществляется; эти модули содержат большое количество команд, не имеющих никакого отношения к логике работы программы.)

2. изменение защищаемой программы (заключается в изменении начала защищаемой программы таким образом, чтобы стандартный дизассемблер не смог ее правильно дизассемблировать.)

Методы противодействия динамическим способам снятия защиты программ от копирования

- периодический подсчет контрольной суммы, занимаемой образом задачи области оперативной памяти, в процессе выполнения. Это позволяет: - заметить изменения, внесенные в загрузочный модуль; - в случае если программу пытаются «раздеть», выявить контрольные точки, установленные отладчиком;
- проверка количества свободной памяти и сравнение и с тем объемом, к которому задача «привыкла» или «приучена». Это действия позволит застраховаться от слишком грубой слежки за программой с помощью резидентных модулей;

- проверка содержимого незадействованных для решения защищаемой программы областей памяти, которые не попадают под общее распределение оперативной памяти, доступной для программиста, что позволяет добиться «монопольного» режима работы программы;
- проверка содержимого векторов прерываний на наличие тех значений, к которым задача «приучена». Иногда бывает полезным сравнение первых команд операционной системы, отрабатывающих этим прерывания, с теми командами, которые там должны быть. Вместе с предварительной очисткой оперативной памяти проверка векторов прерываний и их принудительное восстановление позволяет избавиться от большинства присутствующих в памяти резидентных программ;

- переустановка векторов прерываний; содержимое некоторых векторов прерываний копируется в область свободных векторов; соответственно изменяются и обращения к прерываниям; при этом слежение за известными векторами не даст желаемого результата.
- постоянное чередование команд разрешения и запрещения прерывания, что затрудняет установку отладчиком контрольных точек;
- контроль времени выполнения отдельных частей программы, что позволяет выявить «остановы» в теле исполняемого модуля.