надежности. При этом использование возможностей мандатной модели в большинстве случаев ограничивается рамками операционной системы или доверенной вычислительной сети.

На сегодняшний день существует ряд задач, решение которых выходит за рамки локальной ЭВМ. Среди таких задач можно выделить реализацию следующих технологий:

- сетевые системы управления базами данных;
- безопасная виртуальная консолидация данных;
- защищенная электронная почта;
- службы каталогов (организация принципа Single-Sign-On);
- сетевая печать (маркировка документов в соответствии с мандатным уровнем пользователя);
- конференц-связь.

Для решения данных задач предлагается построение защищенного домена на базе модифицированной ОС Linux с учетом мандатного контроля доступа. В рамках создания домена подразумевается реализация следующих особенностей:

1. организация знания и доступа к сетевой базе учетных атрибутов пользователей (в том числе и мандатных);

2. организация распространения информации о доступных сетевых сервисах;

3. организация аутентификации на удаленных сетевых сервисах;

4. разграничение доступа маршрутизации сетевого трафика по признакам принадлежности информации, установленным ОС. Использование мультипротокольной коммутации по меткам (MPLS) [1, 2] позволяет гарантировать качество обслуживания при передаче больших потоков информации.

Поддержка реализации приведенных выше свойств домена требует расширения соответствующих протоколов передачи данных и маршрутизации, которая выполняется таким образом, чтобы сохранить совместимость с немодифицированными системами.

Создание домена защищенных операционных систем существенно расширяет область их применения по обработке информации, позволяя проводить распределенную обработку ресурсов при сохранении мандатного механизма доступа к данным.

СПИСОК ЛИТЕРАТУРЫ:

1. *Гольдштейн А. Б.* Исследование механизма туннелирования мультимедийного трафика в сети MPLS. Дисс. … канд. техн. наук. СПб., 2004.
2. *Гольдштейн А. Б, Гольдштейн Б. С.* Технология и протоколы MPLS. СПб.: БХВ-Санкт-Петербург, 2005.

А. А. Краснопевцев

О ЗАЩИТЕ ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ ВНЕШНЕГО АППАРАТНОГО МОДУЛЯ

В рамках работы проводилось исследование различных способов защиты программного обеспечения от несанкционированного копирования. В результате проведенного анализа для приложений, компилируемых в некоторое промежуточное представление, т. е. выполняемых на некоторой виртуальной машине, а не на реальном процессоре, было выявлено, что применение существующих

программных способов защиты приложений нецелесообразно. Неприменимость подобных методов защиты связана с тем, что стойкость такого рода систем защиты к компрометации крайне низка. Обычно для этого класса систем защиты удается построить алгоритм, который за линейное время «снимает» систему защиты, что, в свою очередь, ведет к возможности совершить несанкционированное копирование приложения и его запуск [1].

Кроме того, в результате анализа было выявлено, что предложенные методы защиты подобного класса приложений с использованием некоторой аппаратной платформы (обычно электронных ключей различного принципа действия) также имеют свои, достаточно серьезные недостатки [1, 2]. При анализе аппаратных систем защиты приложений было установлено, что все выявленные недостатки можно условно разделить на три класса:

- относительная простота компрометации и построения автоматизированной или автоматической системы компрометации системы защиты;
- сильное снижение скорости выполнения защищенного приложения;
- недоказуемость стойкости системы защиты и невозможность установить желаемый уровень стойкости.

В результате выявленных недостатков были сформулированы требования к методам защиты приложений, которые не имели бы подобного рода недостатков. Основные из них приводятся ниже [3, 4]:

- наличие строгого доказательства стойкости предложенной системы защиты приложений от копирования;
- относительно небольшое снижение скорости выполнения защищенного приложения в сравнении с его незащищенным аналогом;
- процесс компрометации предлагаемой системы защиты должен быть вычислительно сложной задачей;
- сложность восстановления оригинального приложения из его защищенной копии должна основываться на некотором ключевом материале для обеспечения строгого доказательства стойкости системы защиты приложений.

В результате сформулированного набора требований, предъявляемых к системе защиты, был предложен способ защиты приложений с использованием внешнего аппаратного модуля.

Суть предлагаемого в данной работе метода защиты заключается в том, что часть приложения выполняется на оригинальной, хостовой машине, а часть кода — на процессоре внешнего аппаратного модуля, а именно электронного ключа с загружаемым кодом.

Весь процесс защиты можно разделить на несколько этапов.

Первый этап защиты приложений состоит в том, что приложение представляется в виде своего графа потока передачи управления. Данный этап необходим потому, что в дальнейшем вся интерпретация приложения, а также обоснование стойкости предложенной системы защиты будут базироваться на сложности восстановления графа оригинального приложения по графу защищенного приложения.

Далее по полученному графу потока управления защищаемого приложения строится конечный автомат.

Перед построением автомата необходимо задать его. В рамках данной работы используется не стандартный тип автомата Мили, а его расширенная версия $A=\{X,S,Y,S_0,S_e,h,f\}$. В отличие от классического конечного автомата, в рассматриваемом в данной работе имеется два дополнительных множества, а именно S_0 и S_e — множества начальных и конечных состояний, т. е. таких состояний, в которых приложение, представленное автоматом A, может находиться либо в начале своего функционирования, либо непосредственно перед корректным прекращением работы.

При построении автомата используются следующие правила.

Множество состояний автомата A — виртуальные метки, которые ставятся в точках ветвления защищаемого приложения.

Множество входов автомата — переменные, в соответствии со значением которых происходит тот или иной переход по графу потока управления защищаемого приложения.

Множество выходов — код, который необходимо выполнить для того, чтобы перейти из одного состояния автомата в другое.

Функции переходов и выходов данного автомата — табличное представление связей вершин в графе потока управления.

В результате над защищаемым приложением строится конечный автомат, который полностью описывает логику переходов данного приложения.

В дальнейшем алгоритм защиты приложения сводится к тому, что множество состояний автомата, интерпретирующего защищаемое приложение, переносится во внешний аппаратный модуль.

Путем осуществления подобного набора преобразований над кодом приложения злоумышленник для анализа получит набор вершин графа потока управления. В случае, если вершины графа никак не связаны и невозможно выделить некоторые характеристики вершин, позволяющие восстанавливать дуги графа, то сложность восстановления графа будет составлять $O(2^{n^2})$, где n число вершин графа.

СПИСОК ЛИТЕРАТУРЫ:

1. *Чернов А. В.* Анализ запутывающих преобразований программ. URL: http://citforum.ru/ security/articles/analysis/.
2. *Краснопевцев А. А.* Разработка средств автоматизации для переноса байт-кода во внешний аппаратный модуль // Технологии Microsoft в теории и практике. М.: Вузовская книга, 2008. С. 139–141.
3. *Холанд Г., Мак-Гроу Г.* Взлом программного обеспечения, анализ и использование кода. М.: Издательский дом «Вильямс», 2005. – 400 с.: илл.
4. *Букасов В. А, Краснопевцев А. А.* Автоматизация динамической распаковки программ // Материалы 10-й Международной научно-практической конференции. Таганрог. 24–27 июня 2008 г. С. 13–14.

С. В. Ктитров, А. А. Кокуев, Р. Г. Козин

РЕАЛИЗАЦИЯ МЕХАНИЗМА УСТАНОВКИ ПРАВ ПРОЦЕССА ПРИ ЗАПУСКЕ В ОПЕРАЦИОННЫХ СИСТЕМАХ MICROSOFT WINDOWS XP и WINDOWS 7

Требование защиты данных пользователя и сохранения целостности и работоспособности операционной системы приводит к необходимости разделения полномочий пользователей, а следовательно, и запускаемых ими процессов на привилегированные, имеющие административные полномочия, и пользовательские. В ряде ситуаций такое деление не позволяет обеспечить надлежащую защиту данных даже при реализации списков управления доступом.

Рассмотрим задачу передачи данных между пользователями. Предположим, пользователь a наделен привилегиями $R(a)$, пользователь b имеет привилегии $R(b)$, S — административные привилегии, причем $S \cap R(a) = \varnothing$; $S \cap R(b) = \varnothing$; $R(a) \cap R(b) = C$. Если $C \neq \varnothing$, возможен обмен данными между пользователями a и b как через подмножество файловой системы, так и с