УДК. 004.94

В. А. Десницкий, И. В. Котенко

КОМБИНИРОВАННАЯ ЗАЩИТА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ОТ НЕСАНКЦИОНИРОВАННЫХ ВОЗДЕЙСТВИЙ

Рассматривается подход к построению комбинированного механизма защиты программ от несанкционированных воздействий на основе принципов „удаленного доверия“. Предложенный подход обеспечивает достижение компромисса между суммарным уровнем защиты и масштабируемостью механизма.

Ключевые слова: *защита программ, несанкционированные модификации, принципы „удаленного доверия“, атаки на программы и защита от них.*

Введение. Проблема защиты программного обеспечения от несанкционированных воздействий (изменений, модификаций) является одной из наиболее важных и актуальных проблем в области информационной безопасности. В настоящее время наблюдается все большее число несанкционированных модификаций, совершаемых злоумышленниками, с последующим неправомерным использованием программ. К подобным программам, требующим защиты от модификаций, может быть отнесен широкий спектр программных приложений, таких как корпоративные программы и системы документооборота предприятия, пакеты офисных программ, операционные системы, сетевые и коммуникационные программы, а также отдельные программные компоненты, встраиваемые в операционные системы, текстовые и графические редакторы, интегрированные среды разработки, компиляторы. При этом в настоящее время не существует полностью надежных и универсальных решений проблемы защиты, позволяющих гарантировать безопасность и правильность работы программы, функционирующей в ненадежном окружении.

При реализации эффективной защиты программного обеспечения (ПО) на основе „клиент-серверной“ архитектуры важнейшим препятствием к ее осуществлению является необходимость приемлемой масштабируемости механизма защиты.

Таким образом, актуальность исследования проблемы защиты ПО от несанкционированных воздействий не вызывает сомнений. Одной из основных задач таких исследований является построение стойкого и масштабируемого механизма защиты, позволяющего гарантировать безопасное и неизменное выполнение большого числа удаленных клиентских программ.

В отличие от известных моделей механизмов защиты, таких как Pioneer [1], SWATT [2], Genuinity [3] и других, осуществляющих защиту ПО на базе „клиент-серверной“ архитектуры и реализующих, в частности, принцип удаленной аттестации, предлагаемая в настоящей статье модель характеризуется динамическим характером защиты, которая описывается следующими свойствами:

— динамическим изменением отдельных методов защиты, используемых в рамках общего механизма защиты: в частности, могут быть изменены применяемые в механизме алгоритмы, криптографические схемы, сетевые протоколы;

— динамической установкой и вводом в действие различных программных модулей используемых методов защиты; установка модулей осуществляется без перезагрузки и приостановки работы механизма защиты в целом и защищаемой им программы.

Выбор конкретных методов осуществляется в зависимости от характеристик их ресурсопотребления (потребления вычислительных ресурсов) и степени их защиты. При этом установка программного модуля защиты выполняется как на клиентской, так и на серверной

сторонах сетевого соединения, что гарантирует непрерывное корректное выполнение процесса защиты и целевых функций программы.

Модель механизма защиты. Разработанная модель механизма защиты ПО от несанкционированных воздействий относится к классу структурно-функциональных моделей и описывает основные структурные элементы механизма, их функции и возникающие между ними взаимосвязи. Модель описывает также процесс управления защитой, на основе которого администратор системы может управлять механизмом защиты в реальном времени.

Основными объектами предлагаемой модели защиты являются прикладная клиентская программа, которая выполняется на ненадежном хосте, и удаленный защищенный сервер, функционирующий на надежном хосте [4]. Объектом защиты является клиентская программа. Основная цель механизма защиты — гарантия корректности и неизменности выполнения клиентской программы, функционирующей в потенциально враждебном программно-аппаратном окружении. Нарушение корректности и неизменности программы может происходить вследствие намеренных воздействий со стороны нарушителя. Требование неизменности программы предполагает отсутствие возможности выполнения нарушителем каких-либо несанкционированных воздействий, которые могут изменить ход ее работы, в результате чего программа перестанет выполняться корректно.

Угроза неизменности выполнения программы связана как непосредственно с несанкционированными модификациями ее кода или программного состояния, так и с влиянием на программу опосредованно — со стороны ее программно-аппаратного окружения. В последнем случае такое влияние осуществляется, например, посредством использования разнообразных отладчиков бинарного кода и эмуляторов [5, 6], что позволяет без каких-либо изменений программного кода или состояния программы изменить характер ее функционирования.

Любой пользователь, выполняющий или пытающийся выполнить какие-либо несанкционированные действия, называется атакующим, а последовательность действий, осуществляемых им для достижения своих целей, — атакой на программу.

Среди возможных атак на механизм защиты можно выделить следующие: процесс обратной разработки (reverse-engineering), модификация кода программы, динамическое изменение состояния программы, воздействие на среду выполнения программы, перехват и замена сетевого трафика между сервером и клиентом. Отдельного рассмотрения заслуживает такой тип атаки, как „клонирование“: в этом случае одновременно выполняются две или более копии программ, при этом предполагается, с одной стороны, выполнение корректной копии программы, с помощью которой осуществляется взаимодействие клиента с доверенным сервером и при необходимости с другими удаленными сетевыми сущностями, а с другой — выполнение несанкционированно измененной копии программы. Таким образом, любое сетевое взаимодействие модифицированной копии целенаправленно прерывается атакующим, а все необходимые данные для ее работы нарушитель получает посредством анализа сетевого трафика, приходящего и исходящего от корректной копии программы.

Принципы „удаленного доверия“. Предлагаемый механизм защиты базируется на следующих основных принципах „удаленного доверия“ [4].

Архитектура „клиент-сервер“. Этот принцип предполагает реализацию механизма защиты с использованием доверенного сервера, функциями которого являются, во-первых, слежение за состоянием клиентской программы и обнаружение каких-либо отклонений, свидетельствующих о нарушениях безопасности (функция обнаружения), и, во-вторых, реагирование на подобные изменения (функция реагирования).

Удаленная аттестация. Этот принцип заключается во внедрении в клиентскую программу специализированного программного модуля — монитора, который собирает данные о ходе выполнения программы и отсылает их для проверки на сторону доверенного сервера. Принцип реализуется посредством применения таких методов, как проверка инвариантов,

проверка контрольных сумм и др. Данный принцип направлен на осуществление сервером функции обнаружения.

Разделение кода (code splitting). Данный принцип состоит в том, что в коде клиентской программы выделяются и переносятся на сторону доверенного сервера определенные фрагменты, являющиеся наиболее критичными по отношению к информационной безопасности. В результате атакующий не имеет прямого доступа к процессу выполнения кода на отдельных участках, а следовательно, не может на него воздействовать. Примером метода защиты, реализующего данный принцип, является метод барьерного разделения (barrier slicing).

Динамическое замещение. Принцип состоит, во-первых, в регулярном замещении монитора его новой версией и, во-вторых, в замещении некоторых наиболее критически важных фрагментов кода программы измененными фрагментами. Такое периодическое обновление в обоих случаях направлено на усложнение осуществления атак на механизм защиты и защищаемую им клиентскую программу.

Требования к механизму защиты. Сформулируем основные требования, предъявляемые к механизму защиты:

— *построение механизма защиты на основе принципов „удаленного доверия“:* реализация этих принципов осуществляется посредством применения различных отдельных методов защиты, каждый из которых включает определенные криптографические и иные алгоритмы защиты и строится на определенных теоретических принципах; разнообразие методов и принципов защиты позволяет повысить стойкость механизма защиты в целом;

— *динамическое изменение совокупности используемых методов защиты:* эффективность той или иной совокупности методов защиты зависит от соотношения объемов ресурсов доверенного сервера и особенностей функционирования клиентских программ; модель механизма защиты включает возможность отслеживания изменений определенных характеристик доверенного сервера, в том числе объема доступных ресурсов, и пересмотра текущей комбинации методов защиты;

— *добавление и установка программных модулей защиты в режиме реального времени:* предполагается возможность удаления из механизма защиты тех методов, которые признаны уязвимыми или для которых существуют более эффективные способы реализации; после корректной установки модуля защиты и необходимых действий по достижению механизмом защиты определенных требований данный метод может быть использован; при выполнении этого требования важным является то, что загрузка программного модуля и его удаление осуществляются без приостановления процесса защиты и перезапуска клиентского и серверного ПО;

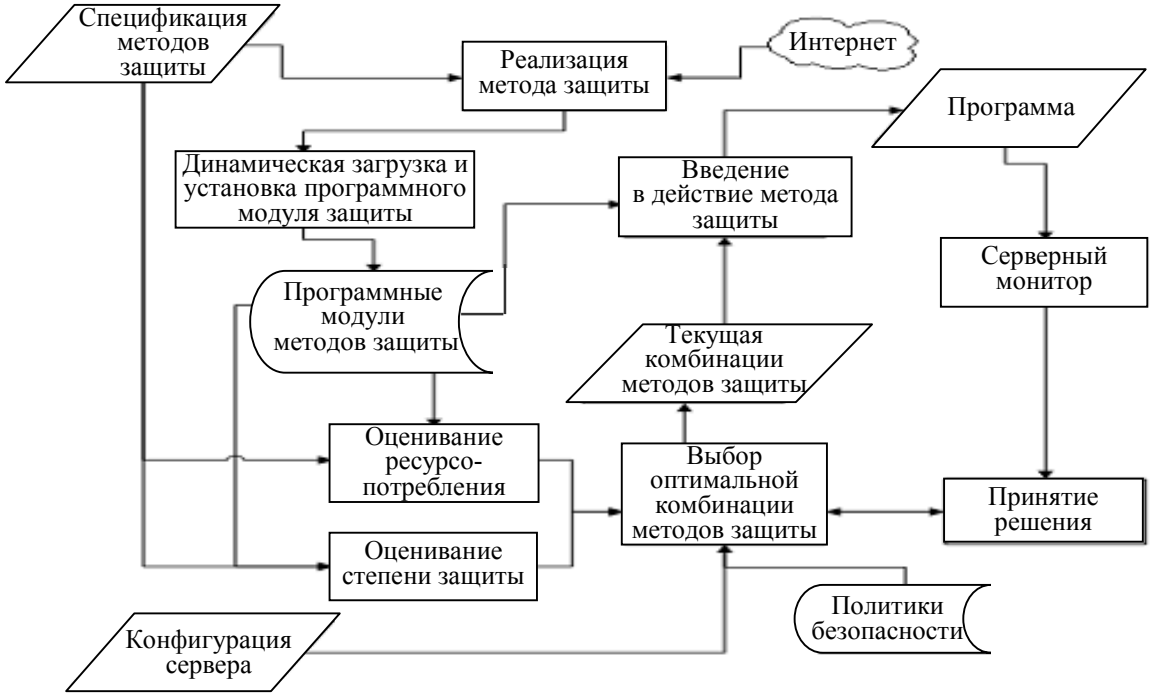
— *дифференциация клиентов:* образование иерархии работающих с клиентскими программами пользователей на основе их ролей, привилегий и мандатов, а также использование политик безопасности как способа задания определенных особенностей механизма защиты; политики безопасности используются также для единообразного управления механизмом защиты.

На рисунке показана структура механизма защиты, демонстрирующая его основные сущности и связи.

Реализация механизма защиты. Реализация механизма защиты сводится к решению следующих задач: оценивание параметров ресурсопотребления каждого из используемых отдельных методов защиты; определение степени защиты, которую способен обеспечить каждый из методов, а также определение суммарной степени защиты механизма в целом; постановка и решение задачи по нахождению оптимальной комбинации методов защиты; обеспечение динамического характера реализуемого механизма защиты.

Оценивание параметров ресурсопотребления происходит эмпирически на основе моделирования отдельных методов защиты и процедур вычисления показателей ресурсопотребления. Определяются следующие показатели: количество $p(m_i)$ фактов использования метода m_i,

исполнение которых доверенный сервер способен поддерживать и контролировать корректно; объем $p^r(m_i)$ ресурса r доверенного сервера, который необходим для реализации метода m_i; суммарный показатель ресурсопотребления $p^r(\tilde{M})$, характеризующий ресурсопотребление комбинации методов защиты $\tilde{M}$. Кроме того, определяется вектор ограничений на ресурсы сервера $C[r]$.



При построении механизма защиты, рассматриваемого в настоящей статье, были реализованы процедуры вычисления следующих показателей ресурсопотребления: $p^{(1)}$ — показатель интенсивности нагрузки центрального процессора (CPU) сервера при реализации метода защиты; $p^{(2)}$ — объем памяти, необходимый для реализации метода защиты на стороне доверенного сервера. Значения показателя $p^{(1)}$ для метода „контроля потока управления“ (Control Flowcheking [7]) программой (m_1), метода проверки инвариантов (m_2), метода барьерного разделения (m_3) и метода ортогонального замещения (m_4) в зависимости от количества клиентов (u) приведены в табл. 1 [5, 6, 8, 9].

Таблица 1

u	$p^{(1)}$, %			
	m_1	m_2	m_3	m_4
10	8,7	7,2	59,9	61,1
20	17,4	14,3	95,6	81,5
25	21,8	37,8	—	92,0
50	43,5	35,7	—	—
100	87,1	71,3	—	—

Оценивание степени защиты, обеспечиваемой каждым из методов, базируется на анализе экспертных мнений, что обусловлено значительной сложностью построения метода оценивания на основе формальных подходов. Процесс оценивания степени защиты включает формирование группы экспертов; формулировку заданий (оценивание каждого из предложенных методов защиты по 10-балльной шкале); опрос экспертов и получение данных от них; математическую обработку полученных данных; представление и анализ результатов.

В табл. 2 приведены полученные в ходе экспериментов, на основе анализа экспертных мнений, усредненные значения степени защиты для некоторых методов [5, 6, 8, 9].

Таблица 2

Метод защиты	Степень защиты
Метод „контроля потока управления“	4,3
Метод проверки инвариантов	3,3
Метод барьерного разделения	9,0
Метод ортогонального замещения	7,8
Метод „crypto guards“*	6,2
Метод непрерывного замещения	7,1
Метод обфускации „непрозрачные предикаты“	1,3

* Примечание: название метода (букв.: криптозащита) приведено в редакции разработчика [6].

Определение оптимальной комбинации методов защиты понимается как достижение наивысшей масштабируемости механизма защиты при заданных уровне ресурсопотребления и степени защиты. Оптимизационная задача формулируется следующим образом:

$$\left.\begin{aligned} &\mathbf{p}(\tilde{M}) \to \max_{\tilde{M} \in 2^M}; \\ &\sum_{m_i \in \tilde{M}} p^{(r)}(m_i) \le C[r] \; \forall r : r \in R; \\ &\sum_{m_i \in \tilde{M}} s(m_i) \ge \hat{s}, \end{aligned}\right\}$$

где s — степень защиты каждого из методов m_i.

Таким образом, данная постановка задачи предполагает максимизацию показателя $\mathbf{p}(\tilde{M})$ на множестве возможных комбинаций методов защиты при ограничении „сверху“ на суммарное значение показателя ресурсопотребления (при $|R|$ неравенств такого вида) и ограничении на суммарную степень $\hat{s}$ защиты. Помимо решения данной задачи методом полного перебора, авторами разрабатывается более быстрый способ решения, предусматривающий применение определенных оптимизаций процесса вычисления, что позволит уменьшить количество комбинаций методов защиты.

Динамический характер механизма защиты реализуется на основе применения средств аспектно-ориентированного программирования, позволяющих проводить динамические изменения программы путем добавления небольших фрагментов кода — аспектов — в ходе ее выполнения [10].

Заключение. Предложен подход к построению механизма защиты ПО от несанкционированных воздействий (модификаций) на основе принципов „удаленного доверия“. Помимо требования поддержки должного уровня предоставляемой защиты разработанный механизм реализует также функции повышения его масштабируемости путем поиска наиболее эффективной комбинации отдельных методов, входящих в механизм защиты. Основным результатом предложенного подхода является архитектура механизма защиты, включающая реализацию предъявляемых к нему основных требований. Представлены детальное формальное описание механизма защиты и принципов „удаленного доверия“, на которых он базируется, а также результаты разработки и реализации более точных методов оценивания ресурсопотребления и степени защиты.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект № 10-01-00826), программы фундаментальных исследований Отделения нанотехнологий и информационных технологий РАН (проект № 3.2), а также при частичной финансовой поддержке, осуществляемой в рамках проектов Евросоюза “SecFutur”, “Massif” и др.

СПИСОК ЛИТЕРАТУРЫ

1. *Seshadri A., Luk M., Shi E.* et al. Pioneer: verifying code integrity and enforcing untampered code execution on legacy systems // Proc. of the 20th ACM Symp. on Operating Systems Principles (SOSP '05). N.Y.: ACM Press, 2005.

2. *Seshadri A., Perrig A., Doorn L.V., Khosla P.* SWATT: Software-based attestation for embedded devices // Proc. of the IEEE Symp. on Security and Privacy. 2004 [Электронный ресурс]: <http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1301329>.

3. *Kennell R., Jamieson L.H.* Establishing the genuinity of remote computer systems // Proc. of the 12th USENIX Security Symp. Washington, DC, 2003.

4. *Десницкий В. А., Котенко И. В.* Модель защиты программного обеспечения на основе механизма „удаленного доверия“ // Изв. вузов. Приборостроение. 2008. Т. 51, № 11. С. 23—30.

5. *Atallah M., Bryant E., Stytz M.* A survey of anti-tamper technologies // J. of Defence Software Engineering. 2004. Nov.

6. *Collberg C., Thomborson C.* Watermarking, tamper-proofing, and obfuscation — tools for software protection // IEEE Computer Society. 2001. Nov.

7. *Ахо А., Сети Р., Ульман Д.* Компиляторы. Принципы, технологии, инструменты. М.: Вильямс, 2003.

8. *Ceccato M., Preda M., Majumdar A., Tonella P.* Remote software protection by orthogonal client replacement // Proc. of the 24th ACM Symp. on Applied Computing (SAC 2009). 2009 [Электронный ресурс]: <http://selab.fbk.eu/ceccato/papers/2009/sac2009.pdf>.

9. *Ceccato M., Preda M., Nagra J.* et al. Barrier slicing for remote software trusting // Proc. of IEEE Intern. Working Conf. on Source Code Analysis and Manipulation (SCAM 2007). Paris, 2007.

10. *Десницкий В. А.* Аспектно-ориентированный подход к реализации механизма мобильного модуля в системе защиты программного обеспечения // Материалы науч.-практ. симп. „Национальные информационные системы и безопасность государства“. М.: ОИТВС РАН, 2007.

Сведения об авторах

Василий Алексеевич Десницкий — СПИИРАН, лаборатория проблем компьютерной безопасности; мл. науч. сотрудник; E-mail: desnitsky@comsec.spb.ru

Игорь Витальевич Котенко — д-р техн. наук, профессор; СПИИРАН, лаборатория проблем компьютерной безопасности; E-mail: ivkote@comsec.spb.ru

Рекомендована СПИИРАН

Поступила в редакцию
09.07.10 г.