ПРОГРАММНО-МАТЕМАТИЧЕСКОЕ ОБЕСПЕЧЕНИЕ СИСТЕМ ЗАЩИТЫ ИНФОРМАЦИИ

УДК 004.056

В. А. Десницкий, И. В. Котенко

МОДЕЛЬ КОНФИГУРИРОВАНИЯ ЗАЩИЩЕННЫХ И ЭНЕРГОЭФФЕКТИВНЫХ ВСТРОЕННЫХ СИСТЕМ

Предложена модель конфигурирования безопасных и энергоэффективных встроенных систем, предназначенная для поиска наиболее эффективных комбинаций компонентов механизма защиты на основе решения оптимизационной задачи.

Ключевые слова: *встроенная система, защита информации, проектирование защищенных встроенных систем, конфигурирование.*

Введение. Одна из актуальных задач в области информационной безопасности — проектирование защищенных информационно-телекоммуникационных систем со встроенными модулями (устройствами). Особенности таких систем связаны, во-первых, с автономностью входящих в их состав устройств и, во-вторых, с ограничениями, накладываемыми на ресурсы устройств, что приводит к их низкой производительности [1, 2].

Проектирование встроенных систем традиционно осуществляется с использованием так называемого компонентного подхода, при котором система защиты строится путем комбинирования отдельных компонентов — алгоритмов и решений, реализующих необходимые функциональные свойства системы защиты. Вместе с тем комбинирование компонентов системы защиты с учетом лишь ее функциональных свойств часто оказывается неэффективным ввиду упомянутых существенных ресурсных ограничений и, как следствие, невозможности реализации полученных решений на практике.

Предлагаемая в настоящей статье конфигурационная модель предназначена для поиска наиболее эффективных комбинаций компонентов механизма защиты на основе решения оптимизационной задачи с учетом нефункциональных свойств встроенной системы и ресурсных ограничений.

Вопросам безопасности встроенных систем посвящено множество публикаций. В частности, в работах [3, 4] перечислены ключевые проблемы в этой области: идентификация пользователей, безопасное хранение данных внутри устройства, устойчивость установленного программного обеспечения к модификациям, безопасный доступ к сети, безопасные сетевые соединения и др. При этом современные механизмы защиты встроенных систем предназначены, в первую очередь, для предоставления защиты против определенного вида угроз [5—7]: так, в работах [4, 8, 9] рассматриваются различные классификации видов угроз для встроенных модулей и типов нарушителей исходя из особенностей и возможностей нарушителя и его окружения [10].

В качестве пути достижения компромисса между защищенностью устройства и его ресурсопотреблением авторы работы [11] рассматривают использование „реконфигурируемых примитивов безопасности“ на основе динамической адаптации архитектуры устройства в зависимости от его состояния и окружения. Предлагаемая адаптация базируется на возможности динамического переключения между несколькими механизмами защиты, встроенными в модуль, и на периодическом обновлении элементов этих механизмов.

В настоящей статье, в отличие от работы [11], предлагается подход к конфигурированию, применяемый непосредственно при проектировании устройства, при этом поиск эффективных относительно производительности устройства решений основывается на выборе компонентов системы защиты исходя из нефункциональных ресурсных требований к устройству и заданных критериев оптимальности.

Задача конфигурирования. Под конфигурацией системы защиты понимается множество ее компонентов, использование которых, во-первых, позволяет реализовать все необходимые функциональные свойства системы защиты, во-вторых, удовлетворить ограничениям на объемы ресурсов, выделяемых для выполнения защитных функций, и, в-третьих, удовлетворить ограничениям на программно-аппаратную совместимость устройства.

Если конфигурация удовлетворяет всем трем условиям, она называется допустимой. Оптимальность понимается в соответствии с заданным критерием, определяемым разработчиком системы в процессе ее проектирования. Процесс конфигурирования включает решение следующих задач на стадии проектирования встроенных модулей: поиск допустимых конфигураций; поиск оптимальных конфигураций; проверка допустимости и оптимальности конфигурации.

Решаемая оптимизационная задача является в общем случае многокритериальной экстремальной задачей с заданным набором ограничений. Ее математическая постановка формулируется с использованием теоретико-множественного представления:

$$goal\_function(non\_functional\_properties(configuration)) \rightarrow min,$$
$$constr(functional\_properties(configuration)),$$
$$constr(non\_functional\_properties(configuration)),$$
$$constr(platf\_compat\_properties(configuration)) .$$

Формулировка задачи содержит задание целевой функции (*goal_function*) на основе количественных значений нефункциональных свойств конфигурации и определение ограничений (*constr*) оптимизационной задачи; при этом рассматриваются как ограничения, накладываемые на функциональные и нефункциональные свойства системы защиты (*functional_properties, non_functional_properties*), так и ограничения на программно-аппаратную совместимость (*platf_compat_properties*). Цель оптимизационной задачи — в соответствии с заданной целевой функцией найти экстремальное значение, представляющее оптимальную конфигурацию.

Моделирование нефункциональных свойств. Построение конфигурационной модели базируется на определениях и методологическом аппарате, предложенных в рамках методологии моделирования MARTE (Modeling and Analysis of Real-Time and Embedded Systems — моделирование и анализ систем реального времени и встроенных систем) [12]. В соответствии с этой методологией в качестве разновидности нефункциональных свойств системы защиты выделяются количественные (quantitative) свойства, которые являются измеримыми.

Количественное свойство, во-первых, характеризуется набором значений (Sample Realizations), которые определяются (измеряются) во время работы устройства, причем измерения могут производиться в рамках экспериментов на реальной системе или на основе программного моделирования. В частности, для циклически детерминированных систем такие значения могут быть получены однократно и „экстраполированы“ на последующие временные циклы.

Во-вторых, количественно нефункциональное свойство характеризуется так называемой функцией измерения (Measure), позволяющей сопоставить набор полученных значений с некоторой числовой величиной. К функциям измерения, например, можно отнести некоторые математические функции: max (максимизация множества), min (минимизация), mean (функция усреднения).

Для количественного определения нефункциональных свойств компонент системы защиты может быть запущен в режиме отладки с использованием средств профилирования, либо процедура оценки свойства может быть непосредственно встроена в приложение. В последнем случае необходимо учитывать побочный эффект данной процедуры и, возможно, корректировать получаемые значения.

В соответствии с методологией MARTE выделяются следующие виды аппаратных ресурсов и соответствующих им нефункциональных свойств:

— вычислительные ресурсы (HW_Computing package); основная характеристика ресурса — opFrequencies, определяющая интервал значений рабочих частот процессора; ресурс характеризуется также величинами MIPS, FLOPS, с помощью которых оценивается количество операций, выполняемых в единицу времени;

— ресурс оперативной памяти (HW_ProcessingMemory package); характеристика — объем памяти и время отклика при доступе к памяти;

— ресурс хранения (HW_StorageManager package); характеристика — объем хранилища;

— ресурс коммуникаций (HW_Communication package); характеристика — пропускная способность канала;

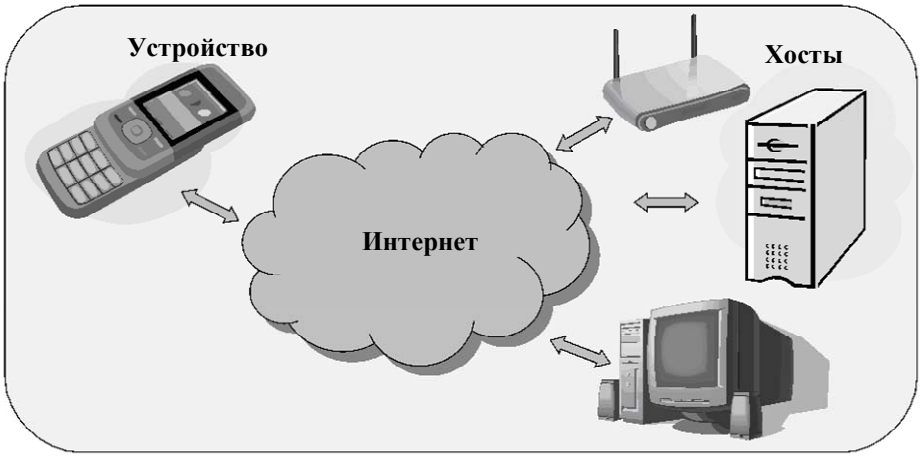
— энергоресурсы (HP_Power package); энергоресурсы расходуются, собственно, на поддержку работы компонентов системы защиты, а также на тепловую энергию; характеристики — мощность источника питания, необходимая для работы устройства (HW_PowerSupply package), и мощность аккумулятора, определяющая продолжительность автономной работы устройства (*HW_Battery package*).

Программная реализация. Авторами разработан программный прототип, реализующий процесс конфигурирования встроенных систем и демонстрирующий принцип конфигурирования. Цель средства конфигурирования — помощь разработчику устройства на стадии его проектирования в принятии решений о выборе наиболее эффективных конфигураций. Разработчику предоставляются: информация о функциональных и нефункциональных свойствах компонентов системы защиты, о свойствах программно-аппаратной совместимости и критериях оптимизации; спецификация защищаемого устройства. В программе также реализованы функции управления процессом конфигурирования. В соответствии с заданными ограничениями и критерием оптимальности, а также множеством компонентов системы защиты программное средство позволяет выявить оптимальные конфигурации. Прототип включает также функцию проверки допустимости и оптимальности заданной конфигурации.

Демонстрационный пример. Продемонстрируем, как принципы конфигурирования могут применяться на практике в процессе проектирования устройств. Демонстрационный пример содержит программные модели мобильного приложения, выполняемого на встроенном модуле, и окружения, в котором модуль функционирует. В частности, в примере демонстрируется ход выполнения методик количественного определения нефункциональных свойств для компонентов и ресурсных ограничений устройства.

Рассматриваемое в примере устройство — смартфон на базе мобильной операционной системы Android (при проведении экспериментов использовалось устройство „HTC Wildfire S“, операционная система Android 3.2), а программное приложение — коммуникационное приложение, функционирующее в сети Интернет („мессенджер“). Задача демонстрационного примера — защита программы от несанкционированных модификаций (см. рисунок).

Ограничимся здесь рассмотрением лишь двух функциональных свойств системы защиты: первое свойство формулируется как „реализация функции постоянного мониторинга внутреннего состояния выполняемой программы посредством принципа удаленной аттестации“; второе свойство — „конфиденциальность критически важных бизнес-данных приложения, хранимых в постоянной памяти устройства“.



В ходе моделирования осуществляется реализация программного приложения, компонентов системы защиты, интегрируемых в него, а также запускаемых в рамках экспериментов функций измерения количественных значений нефункциональных свойств.

Примем следующее допущение. Так как количественные значения нефункциональных свойств измеряются индивидуально для каждого компонента, считаем, что суммарные значения свойств для конфигураций являются суммой соответствующих значений для каждого отдельного компонента. Отметим, что на практике следует учитывать взаимовлияние компонентов системы защиты применительно к расходованию ресурсов устройства. Например, компонент, реализующий удаленную аттестацию, может контролировать целостность не только бизнес-функций программы, но и других компонентов системы защиты.

Рамки моделирования были ограничены рассмотрением трех видов ресурсов устройства как наиболее простых в плане реализации и проведения экспериментов: ресурса оперативной памяти; ресурса хранения; ресурса коммуникаций. Соответственно рассматривались три нефункциональных свойства компонентов: максимальный объем оперативной памяти, расходуемой компонентом на протяжении его работы; минимальный объем постоянной памяти, требуемый для хранения и работы компонента; минимальная пропускная способность коммуникационного канала, необходимая для работы компонента. Таким образом, были заданы и три нефункциональных ограничения для работы устройства: объем оперативной памяти, выделяемый на работу компонентов; объем постоянной памяти, расходуемый на работу компонентов; пропускная способность коммуникационного канала, которую устройство способно предоставить для поддержки работы компонентов.

В ходе экспериментов осуществлялось моделирование двух видов компонентов системы защиты со встроенными в нее процедурами измерения показателей ресурсопотребления. Согласно условиям экспериментов моделируемое приложение содержит некоторое число структур данных, которые рассматриваются в качестве критически важных в контексте защиты от неавторизованных модификаций. Для простоты примем число структур данных одинакового размера — 400 экземпляров байтовых массивов, по 100 байт каждый. Для компонента, реализующего удаленную аттестацию, полагаем, что каждая структура имеет несколько допустимых количественных значений, которые считаются корректными. Если структура имеет отличное от них значение, то это свидетельствует о нарушениях в программе.

Для каждой такой структуры вычисляется хэш-значение (при помощи хэш-функций MD5, SHA), которое отсылается на сторону аттестующего сервера и там подвергается верификации. Значение контрольного интервала задается равным 1 с (т.е. каждую секунду набор подписей — хэш-значений для каждой структуры — должен отсылаться удаленной аттестующей сущности).

Значения расхода ресурсов для каждого компонента приведены в табл. 1 (ресурсы компонентов, реализующих удаленную аттестацию) и в табл. 2 (ресурсы компонентов, реализующих симметричное шифрование критических данных).

Таблица 1

Удаленная аттестация	Расход ресурса		
	памяти, кбайт	хранения, кбайт	коммуникаций, кбит/с
С использованием хэш-функции MD5/128	287	4	51,2
С использованием хэш-функции SHA-256/256	359	4	102,40

Таблица 2

Симметричное шифрование	Расход ресурса		
	памяти, кбайт	хранения, кбайт	коммуникаций, кбит/с
На основе AES/128	523	4	–
На основе DES/56	552	4	–

Использование определенных экспериментально количественных значений нефункциональных свойств позволяет снабдить каждый из компонентов системы защиты набором характеристик его ресурсопотребления. В результате, выбирая наиболее подходящий критерий оптимальности, разработчик определяет оптимальные комбинации компонентов системы защиты (конфигурации являются оптимальными по построению).

При возрастании количества функциональных требований к системе защиты, а также при увеличении числа имеющихся компонентов процедура ручного перебора всех возможных компонентов разработчиком оказывается неэффективной, в частности, вследствие ее экспоненциальной сложности.

Осуществление автоматизированного процесса конфигурирования с использованием разработанной модели позволяет находить оптимальные конфигурации за приемлемое время. В частности, как показали эксперименты, процесс конфигурирования на множестве более чем 100 вариантов компонентов на персональном компьютере выполняется менее чем за 1 с, и даже при увеличении количества компонентов до 1000 и более (если теоретически допустить такую ситуацию) временные затраты окажутся вполне приемлемыми.

Статья подготовлена по результатам исследований, проводимых при финансовой поддержке Российского фонда фундаментальных исследований (проект №10-01-00826-а), программы фундаментальных исследований Отделения нанотехнологий и информационных технологий РАН (проект № 2.2) и государственного контракта 11.519.11.4008, а также при частичной финансовой поддержке, осуществляемой в рамках проектов Евросоюза “SecFutur” и MASSIF.

СПИСОК ЛИТЕРАТУРЫ

1. *Grand J.* Practical secure hardware design for embedded systems // Proc. of the Embedded Systems Conf., San Francisco, CA, CD-ROM, CMP Media, 2004 [Электронный ресурс]: <www.grandideastudio.com/wp-content/uploads/secure_embed_paper.pdf>.

2. *Koopman P.* Embedded system security // IEEE Computer. 2004. Vol. 7, N 37. P. 95—97.

3. *Kocher P., Lee R., Mcgraw G., Ravi S.* Security as a new dimension in embedded system design // Proc. of the 41st Design Automation Conf. DAC ’04. New York, NY, 2004. P. 753—760.

4. *Ruiz J. F., Harjani R., Maña A., Desnitsky V., Kotenko I., Chechulin A.* A methodology for the analysis and modeling of security threats and attacks for systems of embedded components // Proc. of the 20th Euromicro Intern. Conf. on Parallel, Distributed and Network-Based Computing (PDP 2012). Munich, Germany, 15—17 Febr. 2012. P. 261—268 [Электронный ресурс]: <doi.ieeecomputersociety.org/10.1109/PDP.2012.36>.

5. *Десницкий В. А., Чечулин А. А.* Модели процесса построения безопасных встроенных систем // Системы высокой доступности. 2011. № 2. С. 97—101.

6. *Котенко И. В., Десницкий В. А., Чечулин А. А.* Исследование технологии проектирования безопасных встроенных систем в проекте Европейского сообщества SecFutur // Защита информации. Инсайд. 2011. № 3. С. 68—75.

7. *Desnitsky V., Kotenko I., Chechulin A.* An abstract model for embedded systems and intruders // Proc. of the Work in Progress Session Held in Connection with the 19th Euromicro Intern. Conf. on Parallel, Distributed and Network-Based Processing (PDP 2011). Ayia Napa, Cyprus, Febr. 2011. Wilmington, NC: SEA-Publications, 2011. P. 25—26.

8. *Abraham D. G., Dolan G. M., Double G. P., Stevens J. V.* Transaction security system // IBM Systems J. 1991. N 30, Iss. 4. P. 598—598.

9. *Kommerling O., Kuhn M.G.* Design principles for tamper-resistant smartcard processors // Proc. of the USENIX Workshop on Smartcard Technology. CA, 1999. P. 9—20.

10. *Десницкий В. А., Котенко И. В., Чечулин А. А.* Конфигурирование защищенных систем со встроенными и мобильными устройствами // Вопросы защиты информации. 2012. № 2.

11. *Gogniat G., Wolf T., Burleson W.* Reconfigurable security primitive for embedded systems // Proc. of System-on-Chip Intern. Symp., 2005. P. 23—28 [Электронный ресурс]: <http://journals.ohiolink.edu/ejc/article.cgi?issn=19367406&issue=v02i0001&article=1_geitsirsd>.

12. MARTE. The UML Profile for MARTE: Modeling and Analysis of Real-Time and Embedded Systems. Object Management Group, Version 1.1 [Электронный ресурс]: <http://www.omgmarte.org>.

Сведения об авторах

Василий Алексеевич Десницкий — СПИИРАН, лаборатория проблем компьютерной безопасности; 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
10.06.12 г.