ТЕХНОЛОГИЯ ВЫЯВЛЕНИЯ НЕДЕКЛАРИРОВАННЫХ ВОЗМОЖНОСТЕЙ ПРИ СЕРТИФИКАЦИИ ПРОМЫШЛЕННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

***Осовецкий Леонид Георгиевич**, доктор технических наук, профессор,*
г. Санкт-Петербург

Обсуждается проблема совершенствования Российских требований по безопасности в части выявления недекларированных возможностей (НДВ) в функциональном программном обеспечении (ПО). Предлагается подход к получению гарантий по отсутствию программных закладок в ПО при сертификации по требованиям безопасности информации промышленного рограммного обеспечения.

***Ключевые слова**: сертификация ПО по требованиям безопасности информации, технология создания ПО, гарантии по отсутствию НДВ*

DETECTION TECHNOLOGY UNDECLARED CAPABILITIES (NDV) FOR THE CERTIFICATION OF THE SOFTWARE INDUSTRY AT THE REQUEST OF INFORMATION SECURITY

***Leonid Osovetsky**, Doctor of Science (in Tech.)*
Professor, St.Petersburg

The problem of improving the Russian security requirements in terms of detection of undeclared capabilities NDV) in the functionality of the software (SW). An approach to obtain guarantees from the absence of a software program bookmarks with their certification on information security requirements.

***Keywords**: Certification according to the requirements of information security technology for creating software that guarantees the absence of NDV.*

Введение.

Программное обеспечение современных информационных технологий, вычислительной техники и телекоммуникационных систем является сегодня базовой основой для их развития. Рост применения ИТ в различных областях человеческой деятельности и рост влияния ИТ на качественные показатели жизни человека влечет за собой необходимость увеличения объемов разработок ПО, привлечение больших ресурсов на разработку, рост требований по качеству и надежности ПО. Эти проблемы были понятны еще на заре развития отечественного ПО для бортовых вычислительных комплексов[1] и были подробнейшим образом сформулированы В.В. Липаевым [1].

Рост требований и объемов разработки ПО привел к необходимости использования новых технологических решений, в том числе, использование распределенных коллективных усилий с привлечением ранее использованных компонент, жестким технологическим ограничениям для организации коллективной разработки , организацией единой базы разработки, единым управлением разработки, созданием мощных инструментально-технологических средств поддержки разработки.

Технология создания ПО превратилась в многоэтапный инструментально поддержанный регламентированный процесс для своевременного удовлетворения требований потребителя по качеству и надежности ПО.

Поддержка таких технологий под силу только крупным ведущим фирмам разработчикам ПО, которые становятся по сути дела представителями владения стратегическими государственными ресурсами. В этих условиях вопросы безопасности и защиты информационных ресурсов пользователя разработанного ПО становятся в перечне показателей качества на первое место. Это характерно не только для государственного уровня потребителя

1 Макаренко Г.И. Отработка программного обеспечения пилотажно-навигационного оборудования. В кн.: Вычислительные процессы и структуры. Межвузовский сборник, вып.148, ЛИАП, 1981. С. 103-108.

ПО, но и для крупных фирм, банковских структур и малых фирм. В перечне многочисленных показателей уровня безопасности ПО для пользователя (речь идет и о стандартах ОК, BS 7799, американских стандартах, Российских требованиях по безопасности ПО) наряду с традиционными требованиями идентификации, целостности, конфиденциальности, декларируются специальные требования, в том числе требования по контролю отсутствия недекларированных возможностей (программных закладок).

Ввиду изменения технологии создания ПО в сторону использования входящих компонент, организации распределенной удаленной разработки, использования покупных (аутсорсинг) технологий, роль требований по отсутствию недекларированных возможностей возрастает. Будучи руководителем и автором разработки Российского РД (Руководящего документа) ФСТЭК России, получившего название РД НДВ, должен отметить, что в названии этого документа содержатся слова «часть 1», что приводит к частым вопросам: а где же часть2 и что она будет содержать ?

С учетом необходимости использования рассматриваемого РД для сложных промышленных программ, необходимо понимать, что программные закладки возможны на всех технологических этапах разработки программ (а не только при разработке исходных текстов программ), поэтому во второй части РД планировалось изложить методику исследования на содержание НДВ на всех технологических этапах создания ПО, что и было реализовано в ряде сертификаций на отсутствие НДВ. Однако, к сожалению, создание второй части РД НДВ не было поддержано и не финансировалось.

Другой стороной использования при сертификации на отсутствие НДВ является различия между реальным НДВ и уязвимостями, которые выявляют большинство существующих средств, используемых при сертификации на отсутствие НДВ, но без объективного предоставления гарантий на их отсутствие.

Российский руководящий документ по недекларированным возможностям

Российский нормативный документ (РД НДВ)[2] определяющий требования к программному обеспечению в части контроля отсутствия программных закладок разработан в период развития технологии программирования и отражает существующие на тот момент представления о промышленной технологии создания программ. Но ввиду резкого роста объемов разработок ПО и методов сокращения затрат ресурсов на промышленное создание ПО, он не отражает возможные технологические подходы к реализации программных закладок и, соответственно, к технологии их выявления.

Вторая часть должна была определить технологические аспекты выявления НДВ, однако вторая часть не была разработана по не зависящим от разработчиков причинам.

Сам термин «НДВ» - недекларированные возможности, не получил четкого определения в первой части РД, ввиду отсутствия определения «декларированных» возможностей, которые могут быть определены только на основе использования технологических подходов к разработке ПО.

Определение «статистического анализа» исходных текстов программ, основанным на структурном анализе и декомпозиции исходных текстов, не дает четкой методологии структурного анализа, не предусматривает учета технологии создания ПО, которая включает большой перечень формализованных документов и компонент (кроме исходных текстов программ), которые вносят возможность реализации программных закладок и подлежат структурному анализу для корректной методологии их выявления.

«Динамический» анализ исходных текстов программ, предусматривает организацию фактического выполнения всех возможных маршрутов по управлению анализируемого ПО, что практически невыполнимо ввиду огромных объемов современных программных средств и/или выполнимо только при условии использования всех технологических средств разработчика, что организационно практически неосуществимо, так как технология создания ПО является стратегическим ресурсом разработчика и может быть предоставлена только на технологическом стенде разработчика.

РД требует построения (неопределенным методом) всех маршрутов выполнения программ и реализации их выполнения, что для современных объемов ПО требует практически нереализуемых временных и технологических ресурсов.

Требования фиксации контрольных сумм исходных текстов и загрузочных модулей программ не учитывают наличия других технологических компонент, которые включают в современной технологии создания ПО до нескольких сотен опера-

2 Руководящий документ ФСТЭК России «Защита от несанкционированного доступа к информации. Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей.

ций и технологических объектов и которые вносят реальную вероятность наличия недекларированных возможностей, вносимых на любых этапах разработки ПО и, зачастую, носят технологически обоснованный характер, а не вредоносную программную закладку.

В РД не определено понятие «критического» маршрута и методология его выявления.

Требования «динамического» выполнения маршрутов программ с необходимостью встраивания в анализируемое ПО контрольных точек и операций невыполнимо для больших классов современного ПО, которое встроено или прошито в памяти современных вычислительных систем и не подлежит модификации и изменениям, что особенно характерно для бортовых вычислительных комплексов.

Технология получения гарантий по отсутствию НДВ в крупных комплексах программ

На рисунке 1 приведен фрагмент технологии создания промышленных программных средств на примере технологии Rational фирмы IBM.

Эта многоэтапная технология включает фиксацию формализованных «артефактов», контролируемых в технологическом процессе на полноту и непротиворечивость. Формализованные артефакты могут быть представлены в виде иерархической системы таблиц покрытий, которые в этом случае являются формализованной декларацией функций разрабатываемого ПО.

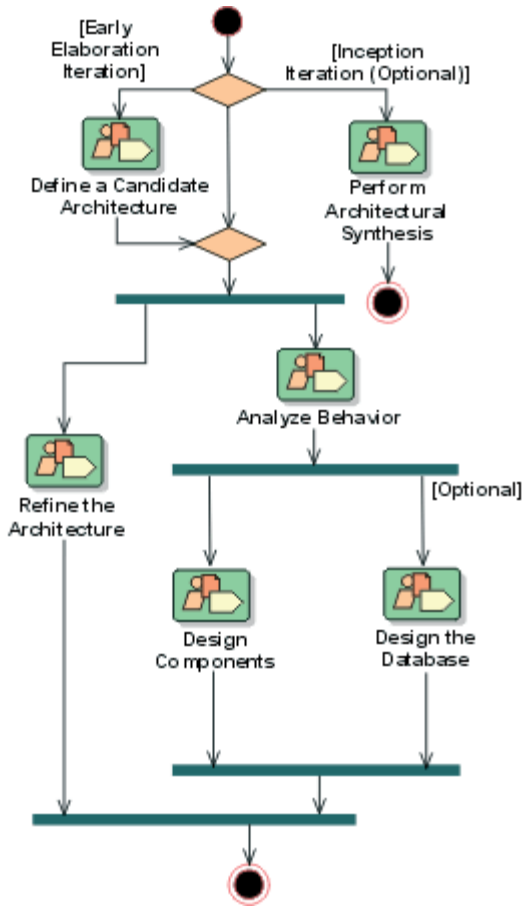

***Рис. 1**. Пример промышленной технологии создания ПО.*

Представление компонент ПО в таком виде позволяет проводить формализованное выявление недекларированных возможностей без использования фактического тестирования, встраивания контрольных точек и операций, то есть и в программных комплексах, «прошитых» во встроенных вычислительных системах.

Проект автоматизированной методики и комплекса ФОБОС на основе применения аппарата комплексных кубических покрытий и графо-аналитических моделей

Научно-технические аспекты процедуры сертификации по Российским требованиям безопасности информации программных продуктов (РД ФСТЭК России) предусматривают наличие обоснованных гарантий по отсутствию недекларированных возможностей (НДВ). В тоже время, имеющиеся средства выявления НДВ и антивирусной защиты не дают высоких гарантий по их отсутствию, что заставляет проводить исследования по методам их гарантированного формализованного выявления и диагностики.

В данной статья предлагается принципиально новая методика сертификации программных средств с использованием графо-аналитической модели и комплексных кубических покрытий.

Любая программа реализует тот или иной вычислительный процесс, который порождается путём интерпретации её команд процессором – реальным или виртуальным. Переход от программ к вычислительным процессам позволяет осуществить поиск решений при сертификации программ в общем виде.

Описанием вычислительного процесса может служить его графо-аналитическая модель (ГАМ) и её описание в виде комплексных кубических покрытий [2-4]

ГАМ можно построить либо путём дешифрации машинного кода программы на основе системы команд процессора с использованием метода структурирования, либо построением её по техническому заданию (спецификации программы). Формализация построения ГАМ программы и её комплексного покрытия базируется на концептуальной двухконтурной итерационно-рекурсивной модели вычислительного процесса, порож-

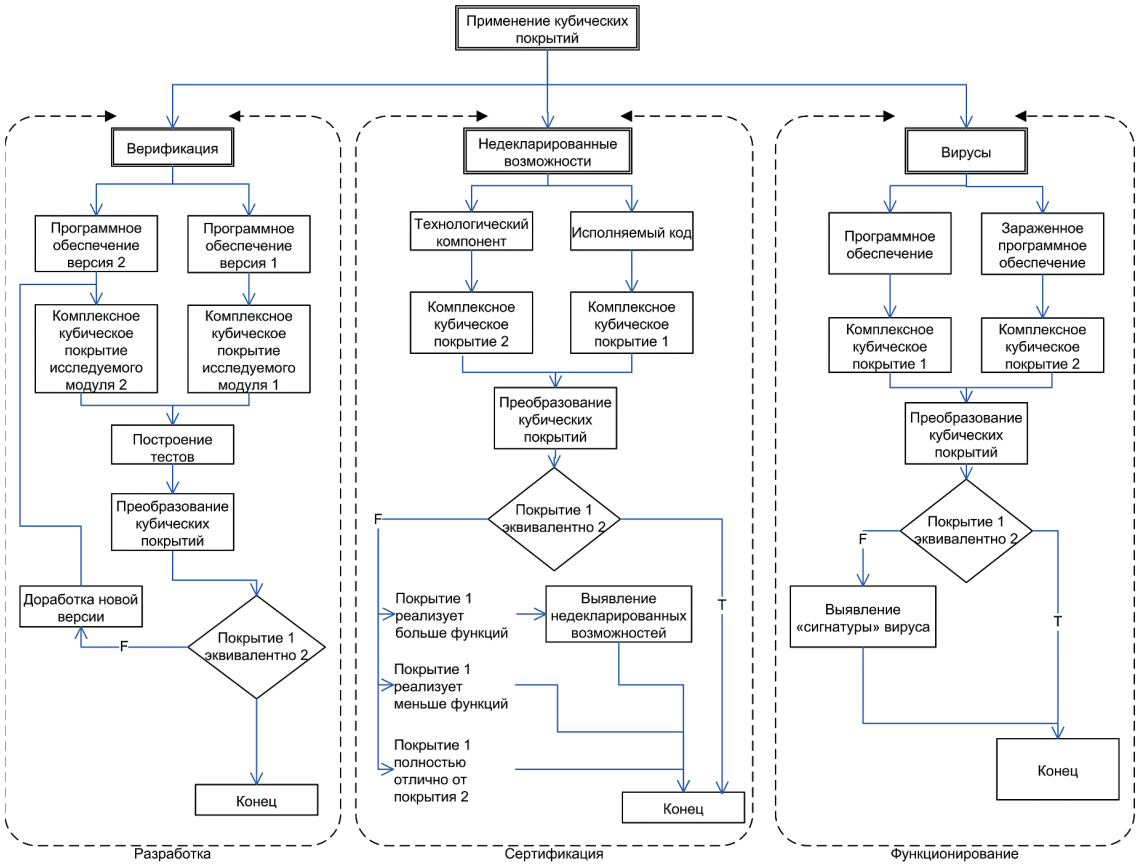

***Рис. 2** Применение кубических покрытий*

даемого процессором при интерпретации команд программы. ГАМ представляется в виде множества вершин и связей (дуг) между вершинами. Вершины могут быть двух типов: линейными, в которых осуществляются вычисления по итеративным и рекуррентным формулам, и условными, в которых вычисляются значения условий-предикатов (УП). Связи являются направленными и осуществляют переход от вершины к вершине с помощью булевых функций управления. Комплексное покрытие объединяет в себе функции управления, логические переменные условий-предикатов и алгебраические выражения вычисляемых переменных. Покрытия строятся по ГАМ методом перебора путей на графе в виде последовательности линейных и условных вершин и связей между ними.

Таким образом, предложенная двухконтурная итерационно-рекурсивная модель вычислительного процесса программ, ГАМ на её основе и построение комплексных кубических покрытий, объединяющих в себе булевы функции, переменные и алгебраические выражения, позволяет перейти от анализа программ к анализу вычислительных процессов. Это дает возможность построить автоматизированные системы для решения задач верификации и тестирования программ, поиска недекларированных возможностей, средств защиты и анализа вирусной опасности[3].

Область применения предлагаемой методики верификации можно разделить на три направления (Рис. 2).

Верификация

Под верификацией понимается установление соответствия между различными программами. Необходимость в этом действии может возникнуть по следующим причинам:

Иногда программный продукт может разрабатываться различными группами проектантов. В этом случае можно верифицировать различные варианты реализаций между собой для повышения объективности и качества выбора конкретной версии;

в ходе жизненного цикла программа может подвергаться различным модификациям: обновление версии или устранение ошибки. Верификация в данном случае позволяет зафиксировать наличие и/или устранение ошибки, подтвердить идентичность логики программ, установить измененные элементы логики.

3 Такая модель позволяет в том числе решать задачи оптимизации программы с точки зрения быстродействия. Прим. ред.

Выявление недекларированных возможностей и мертвого кода

В данном направлении ведется сравнение кубического покрытия построенного по исполняемому коду программы с кубическим покрытием, построенным на основе ее полной спецификации.

Основной задачей исследования вычислительного процесса является его верификация в соответствии с декларацией и состоит в верификации декларированных и недекларированных его возможностей и в поиске несуществующих значений, образующих множество don`t care.

Таким образом, задача верификации вычислительного процесса может быть сведена к поиску НДВ и конъюнкций условий-предикатов, тождественно равных нулю, для которых системы неравенств не имеют решений. НДВ на графе вычислительного процесса образуют множество вершин и дуг, недостижимых через входные последовательности наборов, построенных по декларации и don`t care, которые порождают частично-определенные булевы функции. Эти функции при их отображении на n-мерный двоичный куб могут быть заданы покрытиями комплекса , где f=1 (декларированные возможности, определенные спецификацией), $K^0(f)$, где f=0 (НДВ) и где f=d (don`t care). В случае если верифицируемая программа в точности реализует логику, заданную спецификацией, покрытие $K^0(f)$ будет эквивалентно покрытию, построенному по спецификации, а покрытия $K^0(f)$ и $K^d(f)$ равны пустому множеству. При отображение функций, заданных спецификацией, на n-мерный двоичный куб D_n^2, они могут быть заданы покрытием $K^1(f)$. Таким образом, мы получаем равенство, позволяющее нам верифицировать логику проверяемой программы:

$$E_n^2 - D_n^2 = K^0(f) \cup K^d(f)$$

Если правая часть равенства эквивалентна пустому множеству, то можно делать вывод о том, что логика проверяемой программы в точности соответствует логике заложенной в спецификации. В противном случае проводится дальнейший анализ покрытий $K^0(f)$ и $K^d(f)$.

Исследование вирусов

Как и при поиске недекларированных возможностей, исследуемым объектом является логика работы программ. Предполагается использовать разницу кубических покрытий зараженного экземпляра программы и «чистого». Найденная разница отображает логику вируса. В дополнении с данными о командах обработки данных назовем ее сигнатурой вируса.

Используя подобные сигнатуры можно:

1) исследовать программы на наличие вирусов;

2) исследовать логику работы вируса.

Заключение

В настоящее время основной угрозой в программном обеспечении является наличие недекларированных возможностей. Ни одно из существующих инструментальных средств поиска недекларированных возможностей не дает гарантий их обнаружения. Применение комплексных кубических покрытий позволит не толькогарантированно решить данную задачу, но и автоматизировать этот процесс Кроме этого существуют перспективы использования кубических покрытий в других областях безопасности информации, оптимизации программного кода в части быстродействия и поиска «мертвого» кода.

Литература

1. Липаев В.В. Программная инженерия сложных заказных программных продуктов: Учебное пособие.-М.:МАКС Пресс, 2014.312 с.
2. Немолочнов О.Ф., Зыков А.Г., Поляков В.И., Осовецкий Л.Г., Сидоров А.В., Кулагин В.С. Итерационно-рекурсивная модель вычислительных процессов программ // Известия вузов. Приборостроение». 2005, Том 48, №12. С.14-20.
3. Немолочнов О.Ф., Зыков А.Г., Поляков В.И. Кубические покрытия логических условий вычислительных процессов и программ. // Научно-технический вестник СПбГУ ИТМО. 2004. Выпуск 14. Информационные технологии, вычислительные и управляющие системы. С.225-233.
4. Зыков А.Г., Немолочнов О.Ф. Поляков В.И., Сидоров А.В. Структурирование программ и вычислительных процессов на множество линейных и условных вершин / Научно-технический вестник СПбГУ ИТМО. 2005. Выпуск 19. Программирование, управление и информационные технологии. С.207-212.

References

1. VV Lipaev Software engineering complex custom software: Study posobie.-M.: MAX Press, 2014.-312 p.
2. Nemolochnii O.F., Zykov A.G., Polyakov V.I., Osovetskii L.G., Sidorov A.V., Kulagin V.S Iterative-recursive model of computational processes programs, Proceedings of the universities. Instrument, Volume 48, №12, St. Petersburg, December 2005, pp.14-20.
3. Zykov A.G., Nemolochnii O.F., Polyakov V.I., Cubic coverage logical conditions of computing processes and programs, Scientific and Technical Gazette ITMO. Issue 14. Information technology, computing and control systems, Ch. Ed. VN Vasilev.- St. Petersburg: St. Petersburg State University of Information Technologies, 2004, pp.225-233.
5. Zykov A.G., Nemolochnii O.F., Polyakov V.I., Sidorov A.V., Structuring programs and computing processes on the set of linear and conditional vertices, Scientific and Technical Gazette ITMO. 19. Issue of programming, management and information technology, Ch. Ed. VN Vasilev.- St. Petersburg: St. Petersburg State University of Information Technologies, 2005, pp.207-212.