УДК 004.056

# ПРОГНОЗНАЯ ОЦЕНКА ЗАЩИЩЕННОСТИ АРХИТЕКТУР ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

**А.В. Гвоздев, И.А. Зикратов, И.С. Лебедев, С.В. Лапшин, И.Н. Соловьев**

Предложена методика оценки защищенности архитектур программного обеспечения. Подход позволяет производить сравнительный анализ защищенности программного обеспечения сложных аппаратно-программных комплексов с целью обоснования и выбора их архитектур на ранних этапах жизненного цикла и выявлять наиболее уязвимые программные модули.

**Ключевые слова:** защищенность архитектур программного обеспечения, выбор архитектуры программного обеспечения, оценка уязвимости.

## Введение

Повышение эксплуатационных характеристик программного обеспечения (ПО) подразумевает обеспечение заданных показателей защищенности аппаратной и программной частей. Для оценки защищенности программ используют различные модели, которые разделяют на аналитические и эмпирические [1]. Аналитические модели дают возможность рассчитать количественные показатели защищенности, основываясь на полученных данных о поведении программы в процессе тестирования, эмпирические – базируются на анализе структурных особенностей программных комплексов.

Для предварительной оценки вновь разрабатываемого ПО, анализа компонентного взаимодействия и эмерджентных свойств применяются прогнозные модели, относящиеся к классу эмпирических. Они основаны на измерении технических характеристик создаваемой программы: длина, сложность, число циклов и степень их вложенности, количество ошибок на страницу операторов программы и др. [2]. Учитывая уникальность большинства разработок, их сложность, отсутствие открытой информации по имеющимся аналогам, использование таких данных может быть весьма трудоемким и продолжительным по времени процессом. Зачастую разработчикам ПО требуется на ранних этапах жизненного цикла иметь обоснованное решение об использовании той или иной архитектуры, базирующееся на количественных оценках показателей надежности.

Предлагаемая методика предназначена для получения таких оценок, она дает возможность обосновать структурную схему изделия с точки зрения обеспечения лучших, по сравнению с альтернативными решениями, показателей защищенности, а также выявить наиболее уязвимые программные модули, требующие особого внимания. Важно отметить, что рассматриваемые методы оценки применимы не ко всем типам уязвимостей. Например, они могут быть использованы для оценки защищенности от атак типа «отказ в обслуживании», в то время как оценить, таким образом, защищенность от атак типа «повышение привилегий» практически невозможно.

## Методика оценки

Большинство подходов к оценке защищенности ПО базируется на наличии определенного набора средств и механизмов защиты, позволяющих отнести систему к одному из уровней защищенности, определенному руководящими документами и стандартами. Однако на сегодняшний день все большее значение приобретают количественные оценки, позволяющие производить сравнение и выбор элементов систем. Особенность подхода состоит в том, чтобы рассмотреть влияние программных ошибок на работу с помощью вероятностных показателей, например «вероятности защищенности», на основе которой построить зависимость защищенности функционирования отдельных модулей.

Предлагаемая методика прогнозной оценки защищенности архитектур ПО отличается от известных, основанных на учете качественного состава механизмов и средств защиты, использованием аппарата теории надежности, что позволяет вычислить прогнозные вероятностные количественные показатели защищенности и определить требуемый состав и конфигурацию на ранних этапах проектирования системы. Рассматриваемая методика состоит из следующих шагов:

1. создание функциональной схемы ПО;

2. анализ взаимодействия основных программных модулей;
3. построение эквивалентных схем;
4. определение понятия защищенности;
5. выявление наиболее уязвимых программных модулей;
6. выработка рекомендаций по повышению защищенности программного комплекса.

В зависимости от показателей вероятности защищенности каждого программного модуля в сложных программных системах становится возможным определить влияние отдельных элементов на всю систему в целом для применения комплекса мероприятий (резервирования, тестирования, изменению функциональности и др.) по повышению защищенности.

В случае невозможности тестирования ПО определяются классы ошибок, которые будут оказывать влияние на отказ системы [3]. Классификация таких ошибок в большинстве случаев существенно зависит от функций, выполняемых программным модулем, и требует определения характеристик прогона данных. В общем случае к уязвимостям могут приводить ошибки ПО, возникающие в результате нарушения функционирования системного ПО: сбой операционной системы, коллизий в работе параллельных процессов внутри среды, возникновения ошибки внутри прикладной программы.

Многие из них, при наличии соответствующих систем мониторинга, могут быть устранены путем добавления в модуль диспетчерских функций и резервирования в системе программных процессов, предполагающего использование разных настроек в работе ПО (разные адреса размещения в памяти, требуемая точность вычислений и т.д.) [4].

Функционирование диспетчерского ПО, направленного на повышение надежности системы, является вопросом, требующим отдельного обсуждения, поэтому в настоящей работе основной акцент сделан на использование способа «резервирование».

**Примеры оценки архитектуры ПО с применением резервирования**

Резервирование элементов ПО (например, виртуальных контейнеров, обеспечивающих функции отображения информации на разных рабочих местах) может осуществляться несколькими способами. На рис. 1–3 приведены примеры скользящего резервирования, дублирования линейной последовательности элементов ПО, дублирования элементов ПО в линейной последовательности.

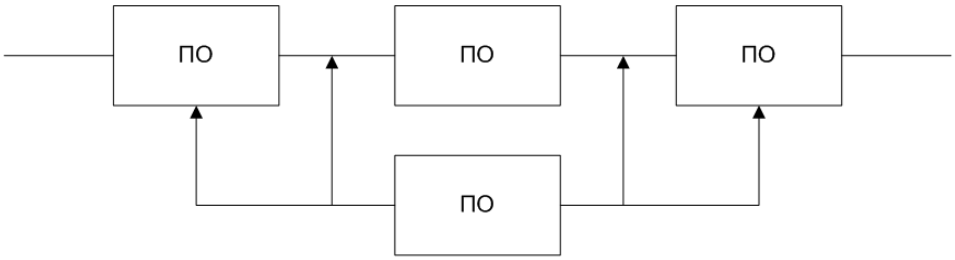


Рис. 1. Скользящее резервирование элементов ПО

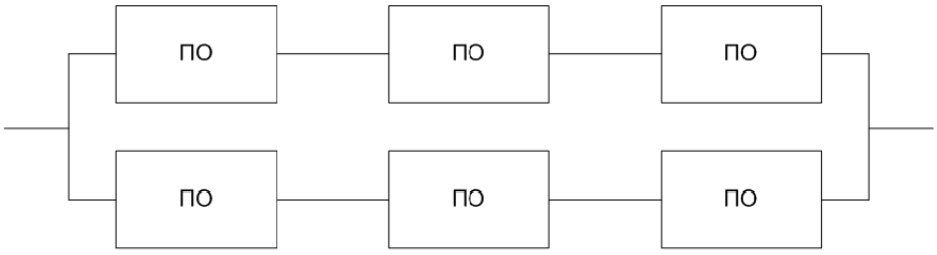


Рис. 2. Дублирование линейной последовательности элементов ПО

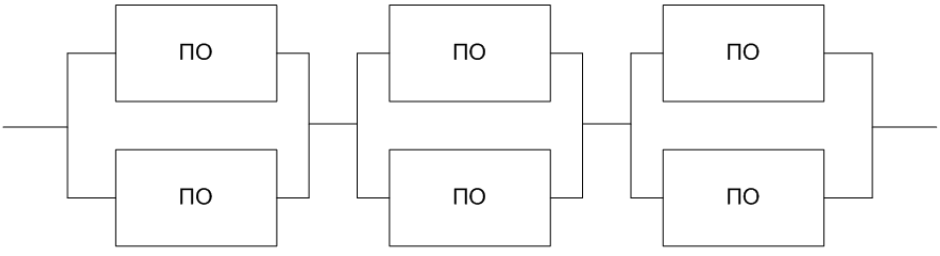


Рис. 3. Поэлементное дублирование в линейной последовательности ПО

Пусть для первого случая, изображенного на рис. 1, для работы программного комплекса системы необходимо $r$ программных элементов, выполняющих функционально одинаковые действия. Для увеличения защищенности ПО создаем $n$ элементов, где $(n–r)$ единиц могут заменить любой из уязвимых элементов последовательности.

Обозначим: $r$ – число необходимых элементов ПО для работы; $n$ – общее число элементов с учетом резервных, которое предполагается создать в системе; $p_0$ – вероятность защищенной работы одного элемента последовательности; $q_0$ – вероятность уязвимости одного элемента последовательности.

Вероятность защищенности $p_0$ каждого элемента оценивается заранее, и может быть получена, например, в результате обработки статистических данных. В общем случае вероятности защищенности каждого из элементов являются независимыми. Тогда вероятность защищенной работы системы $p$ может быть вычислена по формуле [5]

$$p = \sum_{k=r}^{n} C_n^{\,k} p_0^{\,k} q_0^{\,n-k},$$

где $C_n^{\,k} = \dfrac{n!}{(n-k)!k!}$.

На рис. 4 представлены графики зависимостей вероятности защищенности системы от вероятности защищенности одного элемента $p_0$, требующей одновременного функционирования $r$=10 элементов ПО, при общем числе элементов $n$ равных 10 (резервирование отсутствует), 15, 20.

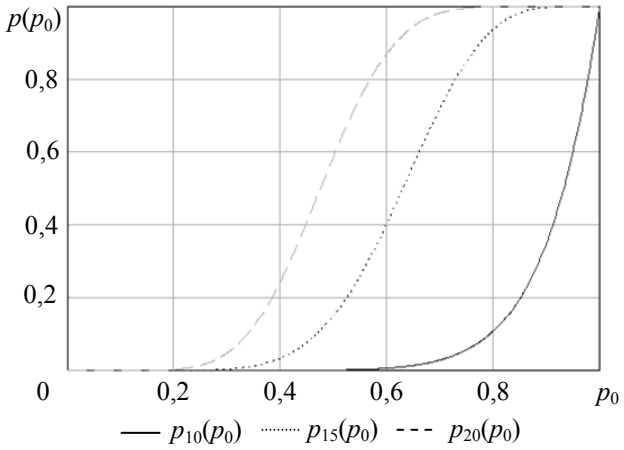


Рис. 4. Зависимость вероятности защищенности системы от вероятности защищенности одного элемента ПО при скользящем резервировании

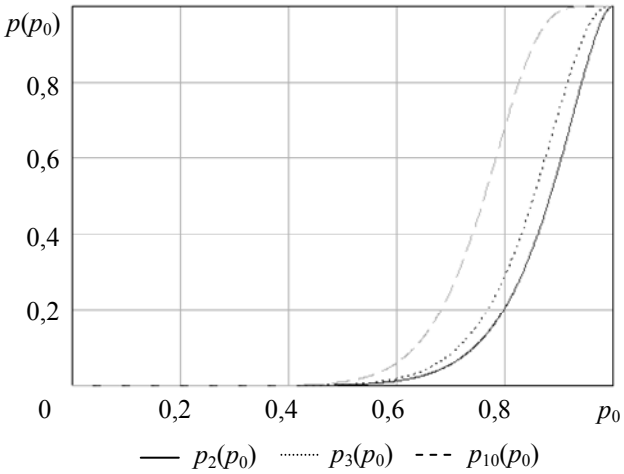


Рис. 5. Зависимость вероятности защищенности системы от вероятности защищенности одного элемента ПО при линейном резервировании

Во втором случае (рис. 2) рассмотрим вероятность защищенной работы программной системы, использующей линейную последовательность элементов. Пусть $n$ – общее число элементов ПО в одной линейке, $p_0$ – вероятность защищенной работы одного элемента последовательности. Вероятность защищенной работы одной линейки из $n$ элементов ПО $p$ определяется соотношением

$p = p_0^n$.

Вероятность защищенной работы дублированной программной системы выражается по формуле:

$p = 1 - (1 - p_0^n)^2$.

В общем случае при резервировании $k$ цепочками выражение примет вид

$p = 1 - (1 - p_0^n)^k$.

На рис. 5 представлены графики зависимостей вероятности защищенности системы от вероятности защищенности одного элемента $p_0$, содержащей число цепочек $k$, равное 2, 3, 10, по 10 элементов в каждой цепочке.

В третьем случае (рис. 3) рассмотрим цепочку из $n$ программных элементов, где дублируется каждый элемент. При вероятности защищенности одного элемента $p_0$ соотношение для системы дублированных элементов определяется формулой [6]

$p = (1 - (1 - p_0)^2)^n$.

При резервировании одного программного элемента не дублированием, а $k$ элементами вероятность защищенной работы выглядит следующим образом:

$p = (1 - (1 - p_0)^k)^n$.

Аналогично предыдущим графикам, на рис. 6 представлена зависимость вероятности защищенности системы от вероятности защищенности одного элемента $p_0$, содержащей $n$=10 элементов, где каждый элемент резервируется $k$ равное 2, 3, 10 элементами.

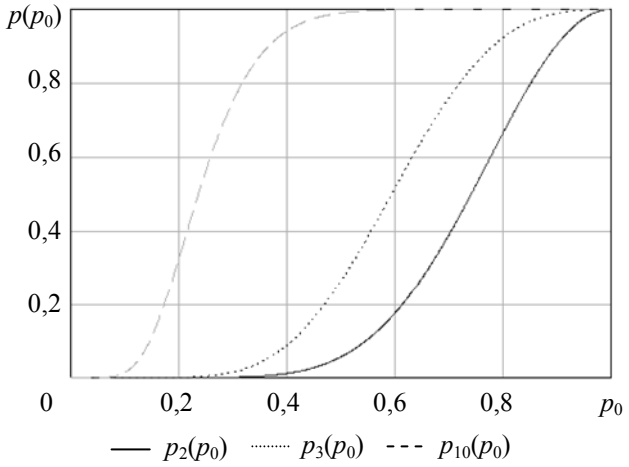


Рис. 6. Зависимость вероятности защищенности системы от вероятности защищенности одного элемента ПО при поэлементном резервировании

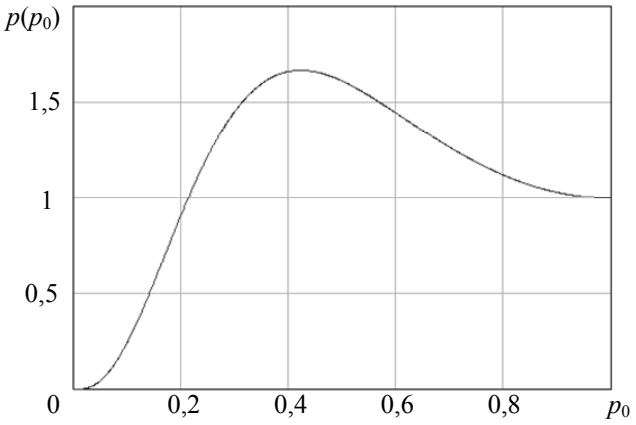


Рис. 7. Зависимость отношения вероятности защищенности различных программных архитектур от вероятности защищенности элемента ПО

Таким образом, предполагая значение вероятности защищенности одного элемента ПО, становится возможным выбрать схему резервирования, исходя из структурных требований к ПО и ресурсоемкости комплекса.

С другой стороны, данный подход позволяет сравнивать архитектуры программных комплексов. Поясним подход на примере. Пусть разработчикам для последнего рассмотренного случая необходимо выбрать структурную схему, позволяющую потенциально достичь наибольшие показатели защищенности. Рассмотрим систему с поэлементным дублированием в линейной последовательности ПО (рис. 3). Допустим, что необходимо обосновать создание последовательности из 3 дублированных элементов ПО, или 6 элементов с четырехкратным резервированием. Для этого составляем отношение

$$p = \frac{(1-(1-p_0)^4)^6}{(1-(1-p_0)^2)^3}.$$

На рис. 7 приведен график отношения вероятности защищенности архитектур. На графике видно, что если ПО имеет вероятность защищенной работы менее 0,2, то решение, содержащее три дублированных элемента, будет предпочтительнее. На отрезке от 0,4 до 1 вторая архитектура превосходит по отношениям вероятности защищенности первую. Максимальный выигрыш от использования второй архитектуры по сравнению с первой – 1,7 раза при вероятности защищенности ПО, равной 0,4–0,5. При стремлении вероятности защищенности программных модулей обеих архитектур к единице отношение также стремится к единице. Физически это можно объяснить тем, что использование абсолютно защищенного ПО приводит к абсолютной защищенности всего ПО независимо от его архитектуры.

В примере рассмотрен упрощенный случай. Аналогичным образом можно поступать и в более сложных ситуациях.

## Заключение

Предложенная методика направлена на оценку защищенности предполагаемых разных архитектур на ранних этапах жизненного цикла программного обеспечения, что позволяет выработать основные принципы и подходы к построению вновь разрабатываемой автоматизированной системе.

Анализ прогнозных результатов расчетов показывает влияние вероятности защищенности каждого элемента программного обеспечения структурных схем, порядка их взаимодействия и количества резервных элементов на вероятность защищенной работы всей системы. Это позволяет выделить наиболее уязвимые элементы программного обеспечения и определить направления повышения защищенности системы.

Работа выполнена в рамках ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007–2013 годы» по государственному контракту № 07.524.12.4009.

## Литература

1. Василенко М.В., Макаров В.А. Модели оценки надежности программного обеспечения // Вестник НГУ. – 2005. – № 30. – С. 126–132.
2. Майерс Г. Надежность программного обеспечения. – М.: Мир, 1980. – 360 с.
3. ISO 9126:1991. Информационная технология. Оценка программного продукта. Характеристики качества и руководство по их применению. – Введ. 01.07.1994. – М.: Изд-во стандартов, 1994. – 18 с.
4. IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Technology (ANSI). – Введ. 28.09.1990. – New York: The Institute of Electrical and Electronic Engineers, 1990. – 84 p.
5. Ушаков И.А. Курс теории надежности систем. – М.: Дрофа, 2008. – С. 27–30.
6. Богатырев В.А., Бибиков С.В. Оценка функциональной безопасности дублированных вычислительных систем // Научно-технический вестник информационных технологий, механики и оптики. – 2012. – № 2 (78). – С. 146.

***Гвоздев Алексей Владимирович*** – Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, аспирант, a.gvozdev@rcwg.net

***Зикратов Игорь Алексеевич*** – Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, доктор технических наук, доцент, зав. кафедрой, zikratov@cit.ifmo.ru

***Лебедев Илья Сергеевич*** – Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, кандидат технических наук, доцент, lebedev@cit.ifmo.ru

***Лапшин Сергей Владимирович*** – Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, аспирант, sv.lapshin@gmail.com

***Соловьев Игорь Николаевич*** – Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, аспирант, sin@inbox.ru