А.С. Марков, В.Л. Цирлов, А.В. Барабанов

Методы оценки несоответствия средств защиты информации

Москва
«Радио и связь»
2012

УДК 621.322
ББК 32.973
М26

Рецензенты:
Академик РАН, д-р техн.наук, проф. *Ю.В. Бородакий*
Член-корр. РАН, д-р техн.наук, проф. *Р.М. Юсупов*

Марков А.С., Цирлов В.Л., Барабанов А.В.
М26 Методы оценки несоответствия средств защиты информации / А.С. Марков, В.Л. Цирлов, А.В. Барабанов; под ред. доц. А. С. Маркова. — М.: Радио и связь, 2012. — 192 с.

ISBN 5-89776-015-2

Книга подготовлена экспертами испытательной лаборатории «Эшелон» как обобщение опыта исследований в области безопасности информационно-программных ресурсов. Содержит метрики, модели и формальные методики испытаний средств защиты информации и тестирования безопасности программного обеспечения, а также актуальные нормативные и правовые требования по сертификации средств защиты информации.

Для специалистов и ученых, увлеченных анализом защищенности, аудитом, испытаниями и тестированием средств защиты информации, программных продуктов и систем в защищенном исполнении.

Совместное издание:
Издательство «Радио и связь»
www.radiosv.ru
и
НПО «Эшелон»
www. cnpo.ru

ISBN 5-89776-015-2

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

АС	Автоматизированная система
АРМ	Автоматизированное рабочее место
ГИР	Государственный информационный ресурс
ЗБ	Задание по безопасности
ИБ	Информационная безопасность
ИСОП	Информационные системы общего пользования
ИСПДн	Информационная система персональных данных
ИТ	Информационная технология
КСЗ	Комплекс средств защиты
МЭ	Межсетевой экран
НДВ	Недекларированная возможность
НСД	Несанкционированный доступ
ОИ	Объект информатизации
ОК	Общие критерии
ОМО	Общая методология оценки
ОО	Объект оценки
ОУД	Оценочный уровень доверия
ПБО	Политика безопасности объекта оценки
ПДИТР	Противодействие иностранным техническим разведкам
ПЗ	Профиль защиты
ПО	Программное обеспечение
ПОК	Потенциально опасная конструкция
ПРД	Правила разграничения доступом
ПС	Программное средство
РД	Руководящий документ
САВЗ	Средство антивирусной защиты

СБФ	Стойкость функции безопасности
СВТ	Средство вычислительной техники
СЗИ	Средство защиты информации
СКЗИ	Средства криптографической защиты информации
СОВ	Система обнаружения вторжений
СП	Сообщения о проблемах
СРД	Система разграничения доступом
СФБ	Стойкость функции безопасности
ТЗ	Техническое задание
ТУ	Технические условия
ФБ	Функция безопасности
ФБО	Функции безопасности объекта оценки
ФТБ	Функциональные требования безопасности

ПРЕДИСЛОВИЕ

«Тестирование программы может эффективно продемонстрировать наличие ошибок, но безнадежно неадекватно для демонстрации их отсутствия».
Дейкстра Э.В. «Смиренный программист»

«А что, при сертификации можно что-то найти?»
Форум руководителей испытательных лабораторий

Идея написания книги связана с опытом авторов в области выявления разного рода дефектов, уязвимостей и угроз безопасности информационно-программных систем и механизмов их защиты. Данный опыт был получен в процессе сертификационных и государственных испытаний, тематических исследований и аудита безопасности более 500 средств защиты информации, продуктов, порталов и систем в защищенном исполнении ведущих зарубежных и отечественных разработчиков.

Необычайный эволюционный рост сложности и динамичности ИТ-продукции показал не только неотвратимость, но и гиперсложность оценки соответствия ИТ-продукции требованиям по безопасности информации. Несмотря на героические усилия ведущих разработчиков, проблема безопасности программных систем не получила своего окончательного решения: число критических уязвимостей не уменьшается, а процесс анализа кода становится чрезвычайно сложной задачей, которую необходимо перманентно решать в рамках периода жизненного цикла программной системы [63, 75, 94]. В этом плане основным механизмом управления информационной безопасностью систем остается сертификация средств защиты информации, эффективность которой в реальной жизни пока зависит от предельной организованности и мозгового штурма экспертов испытательных лабораторий и органов по сертификации. Поэтому применение адекватных методов, метрик и методических приемов может быть весьма полезно, что и является основной целью подготовки монографии.

Кроме факторов технической эволюции, следует отметить необычайный социальный интерес к данной проблеме, отмеченный в нашей стране за последние несколько лет, например, достаточно упомянуть несколько общественных явлений:

— вступление страны в ВТО и неотвратимость исполнения Закона «О персональных данных» глубоко изменили отношение всех юридических лиц страны к защите информации конфиденциального характера со всеми вытекающими последствиями;

— открытая публикация новейших нормативных документов ФСТЭК России, ФСБ России и других регуляторов инициировала всеобщее информирование и внедрение новых современных методологий, методов и средств защиты информации в различных сегментах ИТ-области;

— диалектическое возникновение «сертификационных войн» побудило разработчиков средств защиты к соблюдению правил сертификации на российском рынке компьютерной безопасности и даже раскрытию ведущими западными компаниями (Microsoft, IBM, Oracle, SAP и др.) тайн их исходного кода.

Предлагаемая книга включает 4 главы.

В первой главе дается определение оценки соответствия на основе серии международных стандартов. В ней также описаны процедуры оценки соответствия в области информационной безопасности.

Во второй главе представлено подробное описание понятия сертификации средств защиты информации, ее законодательных и нормативных основ.

Третья глава касается применения математических моделей и методов, которые могут быть использованы при формальных доказательствах результатов испытаний, а также при планировании работ.

В четвертой главе приводятся формализованные методики испытаний средств и механизмов защиты информации по требованиям традиционных и новейших нормативных документов.

Историческая научная база.

Следует сказать, что книга родилась не в вакууме. Среди открытых публикаций советских и российских ученых в области сертификации программ наиболее известны работы п р о ф е с с о р о в Л.Г. Осовецкого [88], В.В. Липаева и А.И.Костогрызова [44], в области оценки надежности программного обеспечения — В.А. Смагина [76], А.Д. Хомоненко [90], Л.В. Уткина [99], статистической теории защиты информации — В.А. Герасименко [22], испытаний комплек-

сов автоматизации — А.С. Шаракшанэ и А.К.Халецкого [86], тестирования подпрограмм — Б.А. Позина [98] и других.

В этом же плане авторы выражают большую благодарность ученым с мировым именем, основательно поддержавшим работу в качестве рецензентов по тематике, а именно: академику РАН Ю.В. Бородакию [16] и член-корр. РАН Р.М. Юсупову [70].

Авторы также признательны за ценные указания и мудрые советы научным кураторам: доцентам В.А. Керножицкому, В.В. Ковалеву, М.М. Котухову и профессору И.А. Шеремету, всем коллегам, активно участвующим в научных мероприятиях, в особенности: доцентам И.В. Зубареву, И.В. Рауткину, А.Ю. Добродееву, В.Г. Маслову, экспертам С.А. Щербине и А.А. Лоскутову, руководству и беззаветным рыцарям науки кафедр «Информационная безопасность» МГТУ им. Н.Э. Баумана и «Программирование» ВКА им. А.Ф. Можайского, всем неутомимым исследователям НПО «Эшелон», а также своим ученым родителям.

Само собой, эта книга была бы невозможна без конструктивного диалога, позитивизма и поддержки со стороны экспертов регулирующих органов и служб. В этой связи авторы выражают самую глубокую признательность руководству и профессионалам федерального и ведомственных органов по сертификации Генерального Штаба ВС РФ и федерального органа по сертификации Федеральной службы по техническому и экспортному контролю.

Монография является результатом коллективного труда.

Помимо авторского коллектива при написании отдельных подразделов участие принимали: В.В. Вареница (подраздел 4.7.1), М.И. Гришин (подраздел 4.4) и А.А.Фадин (подразделы 3.1.3 и 4.7.2).

Авторы также выражают благодарность за ценные рекомендации доценту И.Ю. Шахалову и руководителю испытательной лаборатории М.Ю. Никулину.

Алексей Марков

1. ОСНОВЫ ОЦЕНКИ СООТВЕТСТВИЯ

1.1. Определение оценки соответствия

Под оценкой соответствия понимается доказательство того, что *заданные* требования к продукции, процессу, системе, лицу или органу выполнены[1]. Допускается, что доказательство может быть прямым или косвенным, формальным или неформальным. Выдачу документально оформленного заявления (удостоверения) о соответствии заданным требованиям называют подтверждением соответствия. Примерами таких удостоверений могут быть сертификаты, аттестаты, заключения, выданные официальными органами по оценке соответствия.

Согласно ИСО 17000 базовыми видами деятельности по оценке соответствия являются:

— испытание,
— контроль,
— сертификация,
— аккредитация органов по оценке соответствия.

Виды деятельности могут включать различные процедуры по оценке соответствия. В области информационной безопасности примерами видов деятельности по оценке соответствия или их процедурами являются сертификация средств защиты информации, аттестация объектов информатизации, аккредитация органов, лабораторий, центров, различные виды испытаний и контроля по требованиям безопасности информации, а также аудит безопасности программных средств, информационных систем и систем менеджмента информационной безопасности.

С точки зрения независимости доказательства оценки соответствия различают деятельность трех сторон:

---

[1] ГОСТ Р ИСО/МЭК 17000–2009.

— первая сторона, представляющая объект оценки, например: разработчик или поставщик;

— вторая сторона, заинтересованная в объекте оценки как ее пользователь, например, представитель заказчика;

Таблица 1.1

Виды и процедуры оценки соответствия по требованиям безопасности информации

Виды и процедуры оценки соответствия	**Лицо, организация, орган**		
	Первая сторона	**Вторая сторона**	**Третья сторона**
	Объекты оценки соответствия		
Предварительные испытания	Проекты, макеты, образцы СЗИ		
Входной контроль		СЗИ	
Приемка		СЗИ	
Периодический контроль	СЗИ	СЗИ	СЗИ
Экспертиза			Документы
Сертификация			СЗИ
Аттестация	ОИ*	ОИ*	ОИ
Контроль встраивания			Криптографические СЗИ
Декларирование	Документация		
Поверка			Средства измерений
Калибровка	Средства измерений	Средства измерений	Средства измерений
Спецэкспертиза			Организации
Аккредитация			Организации
Инспекционный контроль			СЗИ, ОИ, организации
Аудит	СЗИ, системы, организации	СЗИ, системы, организации	СЗИ, системы, организации
* только для лицензиатов ФСТЭК при защите конфиденциальной информации СЗИ — средства защиты, ОИ — объекты информатизации			

— третья сторона, независимая от первой и второй сторон, например, аккредитованная испытательная лаборатория.

К примеру, средства защиты информации подлежат оценке соответствия в форме обязательной сертификации. Аккредитованные органы сертификации и испытательные лаборатории, участвующие в процессе сертификации, должны быть независимы от заказчика работ, т.е. являться третьими сторонами. В то же время аттестация объектов информатизации в некоторых случаях может быть проведена самой организацией, т.е. первой стороной. Подтверждение соответствия первой стороной называется декларированием (декларацией) о соответствии. В таблице 1.1 приведены примеры видов деятельности и объекты оценки соответствия по требованиям безопасности информации.

Совокупность участников, правил, процедур и менеджмента, используемых для выполнения оценки соответствия, представляет собой систему оценки соответствия. В нашей стране в области безопасности информации примерами таких систем являются обязательные системы сертификации средств защиты информации Минобороны России (РОСС RU.0001.01ГШ00), СВР России (РОСС RU.0001.01СЗ00), ФСБ России (РОСС RU.0003.01БИ00, РОСС RU.0001.030001) [2] и ФСТЭК России (РОСС RU.0001.01БИ00), а также добровольные системы сертификации по безопасности информации.

Международный стандарт ИСО 17000 определяет три функции оценки соответствия (см.рис.1.1):

1. Выбор, в рамках которого выполняется планирование и подготовка к получению доказательств, в том числе определяется методология оценки и отбирается собственно объект оценки;

2. Определение, в рамках чего проводится исследование характеристик объекта оценки и формируются соответствующие отчетные документы;

3. Итоговая проверка и подтверждение соответствия, в результате которых принимается и оформляется решение о соответствии или несоответствии объекта оценки заданным требованиям.

Важной процедурой оценки соответствия является инспекционный контроль, представляющий собой систематическое наблюдение за деятельностью по оценке соответствия.

---

[2] В области компетенции ФСБ России зарегистрированы две системы сертификации СЗИ (см. www.fsb.ru).

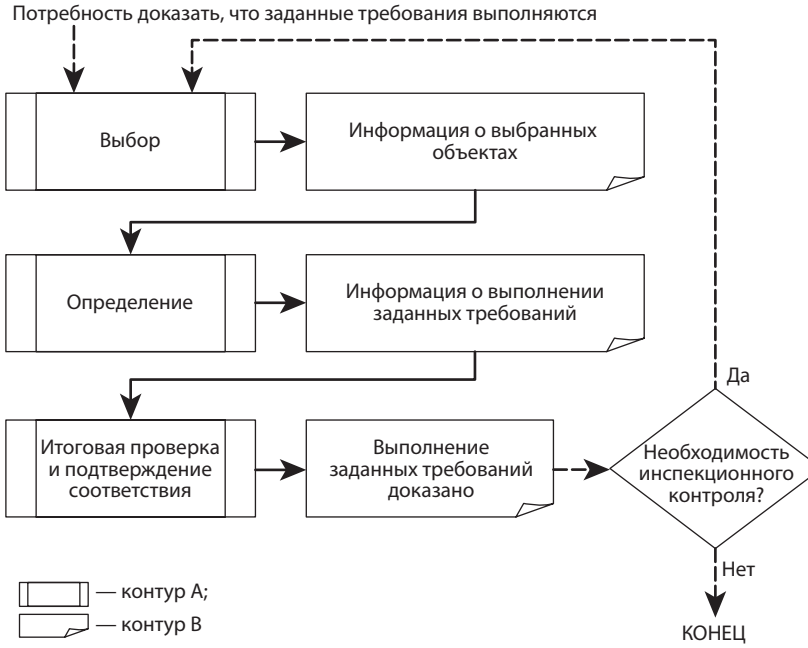

Рис. 1.1. Функциональный подход к оценке соответствия по ИСО 17000

Следует сказать, что область определений и терминов по оценке соответствия имеет весьма размытые рамки. Например, в национальном стандарте ГОСТ 17000–2009 под подтверждением соответствия понимают выдачу документа (заявления) о соответствии или несоответствии, а в Законе № 184-ФЗ «О техническом регулировании» понимают вид деятельности: сертификацию или декларирование соответствия техническому регламенту.

Мы будем придерживаться терминов, используемых в нормативно-методических документах федеральных органов, регулирующих процессы информационной безопасности, если это не будет оговорено отдельно.

1.2. Виды процедур оценки соответствия технических систем

Кроме сертификации средств защиты информации (которая заслуживает отдельного внимания в разделе 2) можно встретить

определения разного рода процедур оценки соответствия технических средств и систем защиты, а именно: испытания, аттестация, тестирование, аудит, анализ рисков.

1.2.1. Испытания

Испытание — вид деятельности или процедура по оценке соответствия, заключающаяся в экспериментальном определении количественных или качественных характеристик объекта испытаний как результата воздействия на него при его функционировании, моделировании или воздействий[3].

Испытания проводятся на основании документа «Программа и методика испытаний», который разрабатывается в испытательной лаборатории и оформляется согласно национальным стандартам (ГОСТ 19.301–79, РД 50–34.698–90, ГОСТ 51719–2001 и др.).

Результаты испытания оформляются в виде протоколов испытаний или технического отчёта.

Как правило, испытания по требованиям безопасности информации проводятся на этапе внедрения. Исключение составляют периодические испытания.

Различают испытания продукции и систем. Например, в ГОСТ 16504 приведено 46 видов испытаний продукции: предварительные, доводочные, периодические, государственные, межведомственные, стендовые, полигонные, натурные, граничные, аттестационные, сертификационные испытания и другие.

Для автоматизированных систем согласно ГОСТ 34.603 (п.1.3.) установлены 3 основных вида испытаний:

1. Предварительные, которые могут быть автономными и комплексными;
2. Опытная эксплуатация;
3. Приемочные.

Национальные стандарты допускают проведение других видов испытаний систем и их составных частей.

В области информационной безопасности требования и виды испытаний устанавливаются уполномоченными государственными регуляторами, в частности, это касается сертификационных испытаний средств защиты информации и аттестационных испытаний объектов информатизации на соответствие ведомственным нормативным документам.

---

[3] ГОСТ 16504–1981.

1.2.2. Аттестационные испытания

Легитимность обработки информации на объектах информатизации подтверждается путем их аттестации, основное содержание которой составляют аттестационные испытания, представляющие собой комплексную проверку защищаемого объекта информатизации в реальных условиях эксплуатации с целью оценки соответствия применяемого комплекса мер и средств защиты требуемому уровню безопасности информации. Положительный результат аттестации оформляется в виде аттестата соответствия.

Наиболее полно регламентирована система аттестации объектов информатизации по требованиям ФСТЭК России. Согласно правилам ФСТЭК России аттестация является частью единой системы сертификации и аттестации и касается именно объектов (систем) [4].

При аттестационных испытаниях подтверждается соответствие объекта информатизации требованиям:

— по защите информации от несанкционированного доступа (в том числе от компьютерных вирусов),

— от утечки за счет побочных электромагнитных излучений и наводок при специальных воздействиях на объект (высокочастотное навязывание и облучение, электромагнитное и радиационное воздействие),

— от утечки или воздействия на нее за счет специальных устройств, встроенных в объекты информатизации.

Проверки, в принципе, зависят от типа объекта информатизации, который может быть одним из следующих:

— автоматизированная система (в том числе сегмент АС [5]);

— помещение, предназначенное для ведения конфиденциальных (секретных) переговоров;

— системы связи, отображения и размножения вместе с помещениями, в которых они установлены, предназначенные для обработки и передачи информации, подлежащей защите.

Надо понимать отличие аттестационных испытаний от сертификационных. Так, аттестационные испытания касаются объектов информатизации (*систем*), проводятся органами по аттестации или лицензиатами ФСТЭК России (при защите конфиденциальной информации), включают выполнение стандартных детерминирован-

[4] Приказы Гостехкомиссии России № № 199 (1995), 3 (1996).

[5] ГОСТ 0043—003—2012.

ных процедур, связанных, главным образом, с фиксацией состава, проверкой легитимности и корректности установки сертифицированных средств защиты информации [24,65,83]. В свою очередь, сертификационные испытания касаются технических *средств* защиты информации, проводятся испытательными лабораториями, включают длительные и трудоемкие процедуры проверки этих средств, в том числе используя методы функционального и структурного тестирования и др.

Особенности аттестационных испытаний регламентируются *специальными* требованиями и рекомендациями, но с Положением по аттестации (Гостехкомиссии России, 1994 г.) можно ознакомиться на сайте ФСТЭК России [69].

1.2.3. Тестирование программных средств

Согласно международным стандартам тестирование — это техническая операция, заключающаяся в определении одной или нескольких характеристик продукта, процесса или услуги по соответствующей процедуре[6]. Что касается тестирования ПО, то его целью является выявление ошибок (дефектов, недостатков) в программной реализации заданных свойств ПО. Особенности современного производства программ подразумевают, что тестирование интегрировано в систему менеджмента качества ПО на всех этапах жизненного цикла.

Вследствие невозможности всеобъемлющей проверки современного ПО, тестирование ПО стало отдельной научно-технической дисциплиной[7]. Мы коснемся только наиболее важных классификационных признаков, необходимых при оценке соответствия, таких как: уровень знаний об исходном коде, методология проверки, структурный уровень проверки, детерминированность тестов, показатели качества и т.п.

По уровню знаний о структуре ПО различают функциональное (по принципу черного ящика) и структурное (по принципу белого ящика) тестирование[8]. Функциональное тестирование использует всевозможные комбинации входных данных, допустимые входным интерфейсом. Если распределение входных данных приближено к реальному процессу эксплуатации, можно оценить уровень

[6] ГОСТ Р ИСО/МЭК 12119–2000.

[7] IEEE Guide to SWEBOK, 2004.

[8] ГОСТ Р ИСО/МЭК 12119–2000.

корректности и надежности функционирования ПО. Для выявления ошибок, связанных с комбинациями редко используемых данных (например, программные закладки), более практичны методы структурного тестирования.

По методологии проверок различают статическое (без выполнения кода) и динамическое (с выполнением кода) тестирование. Основу динамического тестирования составляют *тесты* — наборы входных данных и условий функционирования.

Статическое тестирование подразумевает либо ручной просмотр (инспектирование, ревизия) текста программ, либо автоматизированный статический анализ. В последнем случае условно различают синтаксический анализ свойств программы на ее лексических/синтаксических/семантических моделях, ориентированный на выявление некорректностей кодирования (нефункциональных дефектов) [23,39,40,74], и сигнатурный анализ, ориентированный на поиск фрагментов кода повышенного риска (как правило, программных закладок) [54,71].

Следует уточнить, что в ряде классификаций под понятие тестирование подпадают только динамические методы[9].

Выделяют несколько структурных уровней тестирования, например:

— модульное (компонентное) тестирование, которое позволяет проверить функционирование отдельно взятого элемента системы[10], например: модулей, подпрограмм, программных файлов, драйверов, приложений, веб-приложений, протоколов, программных интерфейсов;

— интеграционное тестирование путем проверки взаимодействия между программными компонентами (например, путем реализации методов «сверху вниз», «снизу вверх», распределения потоков управления и данных[11] и т.д.);

— системное тестирование, которое охватывает целиком всю систему и ее внешние интерфейсы.

Большинство функциональных сбоев должно быть идентифицировано еще на уровне модульных и интеграционных тестов. В свою очередь, системное тестирование обычно фокусируется на

---

9 IEEE Std 829—2008.

10 ANSI/IEEE 1008—1987.

11 IEEE Guide to SWEBOK, 2004.

нефункциональных требованиях – безопасности, производительности, точности, надежности т.п.

По степени детерминированности разделяют стохастическое и детерминированное (экспертное) тестирование и их комбинации. Для стохастического тестирования применяются генераторы тестов в заданных границах и областях входных данных, что позволяет автоматизировать процесс тестирования. Наиболее известной является техника фаззинг-тестирования (fuzzing, fuzz-testing), когда тесты содержат случайные данные, в том числе искаженные и непредусмотренные, получаемые путем псевдослучайной генерации или мутации входных и промежуточных данных.

Надо понимать, что нерегулируемый стохастический процесс приводит к переборам входных данных астрономического порядка [95].

Что касается свойств качества, то известны тестирование корректности, безошибочности, производительности, безопасности информации и др. При проверке свойств безопасности информации (целостности, доступности, конфиденциальности и др.) задаются требования к подсистемам, например, идентификации, аутентификации, разграничения доступом (авторизации), целостности, протоколирования (аудита, журналирования) и др.

Особенностью тестирования безопасности информации является то, что ошибки в ПО могут быть внесены искусственно (например, отладочная информация, недекларированные возможности) или даже злонамеренно (программные закладки). Более того, выявление возможности реализации этих ошибок может быть тоже целенаправленно злонамеренным. Поэтому, кроме функционального тестирования подсистем безопасности на соответствие заданным требованиям, целесообразно проводить структурное тестирование ПО с целью выявления дефектов и уязвимостей, влияющих на безопасность.

С точки зрения отечественной нормативной базы можно выделить следующие техники тестирования [69]:

– функциональное тестирование подсистем безопасности по требованиям, заданным в нормативных документах и документации;

– проверочное и периодическое тестирование подсистем защиты информации, согласно тестовой документации;

– статический анализ отсутствия недекларированных возможностей;

— динамический анализ отсутствия недекларированных возможностей.

В таблице 1.2 приведены методы тестирования, используемые при проверках средств защиты информации.

Таблица 1.2

Методы тестирования средств защиты информации

Метод тестирования	Основные выявляемые дефекты и уязвимости
Верификация	Дефекты проектирования методов управления доступом
Функциональное тестирование	Дефекты реализации функций и ошибки документации
Фаззинг-тестирование	Дефекты реализации интерфейсов данных
Граничное тестирование	Ошибки граничных условий
Нагрузочное тестирование	Ошибки производительности
Стресс-тестирование	Отказ в обслуживании
Профилирование	Недостатки оптимизации кода
Статический семантический анализ	Некорректности кодирования
Статический сигнатурный анализ	Заданные потенциально опасные фрагменты
Статический анализ отсутствия НДВ [69]	«Мертвый код»
Динамический анализ отсутствия НДВ [69]	«Мертвый код»
Мониторинг операционных процессов	Нарушения целостности процессов и ресурсов
Тестирование конфигураций	Ошибки администрирования
Сканирование уязвимостей	Известные опубликованные уязвимости
Тест на проникновение	Известные уязвимости и ошибки конфигурирования
Регрессионное тестирование	Повторные ошибки прошлых версий
Рефакторинг	Недостатки качества кода

1.2.4. Аудит информационной безопасности

Аудит — систематический, независимый и документированный процесс получения записей, фиксирования фактов или другой соответствующей информации и их объективного оценивания с целью установления степени выполнения заданных требований[12]. Основное отличие аудита от сертификации состоит в том, что здесь нет жестких рамок в плане подтверждения соответствия в виде документа государственного образца (сертификата соответствия строго определенным требованиям), нет необходимости привлечения третьей стороны (независимой аккредитованной лаборатории), при этом критерии аудита также могут быть более гибкие, отвечающие реалиям дня.

Аудит может быть внутренним и внешним (во всех смыслах), проводиться на соответствие любым требованиям, сформулированным заинтересованными лицами. Аудит может носить как технический, так и организационно-нормативный характер. В общем смысле аудит включает проведение различных (динамических и статических) форм *тестирования* подсистем и сегментов, анализ документации и других информационных источников (например, бюллетеней), интервьюирование специалистов и т.д.

В области информационной безопасности различают аудит организаций, информационных систем, систем менеджмента и программного кода. К аудиту безопасности информационных систем относят комплексный анализ защищенности, анализ уязвимостей, тесты на проникновение, аудит на соответствие нормативным документам регуляторов по защите различных тайн [14, 29, 33, 45, 62].

В таблице 1.3 приведены нормативные и методические документы, используемые при различных видах аудита ИБ.

1.2.5. Анализ риска информационной безопасности

Если в области оценки надежности и устойчивости технических систем основополагающими понятиями являются ошибки и отказы, то в области информационной безопасности таковыми понятиями являются угрозы. Негативное последствие угрозы безопасности ресурсу оценивается величиной соответствующего *риска*, под которым понимается комбинация вероятности «нежелательного» события и его последствий[13].

---

[12] ГОСТ Р ИСО/МЭК 17000–2009.

[13] ISO/IEC Guide 73:2002.

Таблица 1.3

Нормативно-методические документы, используемые при аудите безопасности

Категории аудита информационной безопасности	Нормативные и методические документы
Аудит системы менеджмента информационной безопасности	ГОСТ 27001, ГОСТ 53647, BS 10012, Cobit, ГОСТ 15.002
Анализ защищенности	ГОСТ 17799
Тесты на проникновение	OSSTMM, ISSAF, NIST 800—42, OWASP Testing Guide, Wireless Penetration Testing Framework, Penetration Testing Framework
Аудит безопасности кода	OWASP Top Ten, Fortify SPK, MITRE CWE.
Аудит на соответствие требованиям стандарта Банка России по информационной безопасности	СТО БР ИББС-1.1
Аудит на соответствие требованиям регуляторов по защите персональных данных	Нормативные и нормативно-методические документы Роскомнадзора, ФСБ России, ФСТЭК России
Аудит безопасности платежных систем	PCI DSS

Следует отметить, что нарушение правил оценки соответствия тоже может быть риском правового и технического характера.

Систематическое использование информации для идентификации источников и оценки величины риска называется *анализом риска*[14]. Методы анализа риска могут быть качественными, полуколичественными и количественными[15]. В качественных методах используются вербальные понятия уровней риска, например: «маленький», «большой», «очень большой» и т.д. В полуколичествен-

[14] Там же.

[15] ISO/IEC 31010:2009.

ных методах используются численные шкалы (линейные или логарифмические). Количественные манипулируют конкретными числовыми единицами. Заметим, что полный количественный анализ не всегда возможен вследствие эргатичности систем информационной безопасности [96], сложности получения представительной статистики и др.

В настоящее время сложилась нормативная база анализа риска в области информационной безопасности в виде ГОСТ Р ИСО/МЭК 27005–2010 «Информационная технология. Методы и средства обеспечения безопасности. Менеджмент риска информационной безопасности», который определяет процессный подход к управлению рисками, включающий в том числе оценку и обработку риска (см. рис.1.2) [61].

Анализ риска включает этапы инвентаризации и категорирования ресурсов, идентификации значимых угроз и уязвимостей [47, 48], а также оценку вероятностей реализации угроз и уязвимостей. Оценивание риска заключается в вычислении риска и собственно оценивании его по заданной *n*-балльной или табличной шкале. Риск, как правило, вычисляется как функция произведения:

$$R_{ij} = p_{ij} k_{ij} C_j ,$$

где: C_j — «стоимость» j-го ресурса, k_{ij} — коэффициент компрометации j-го ресурса при реализации i-й угрозы, p_{ij} — вероятность реализации i-й угрозы безопасности j-го ресурса.

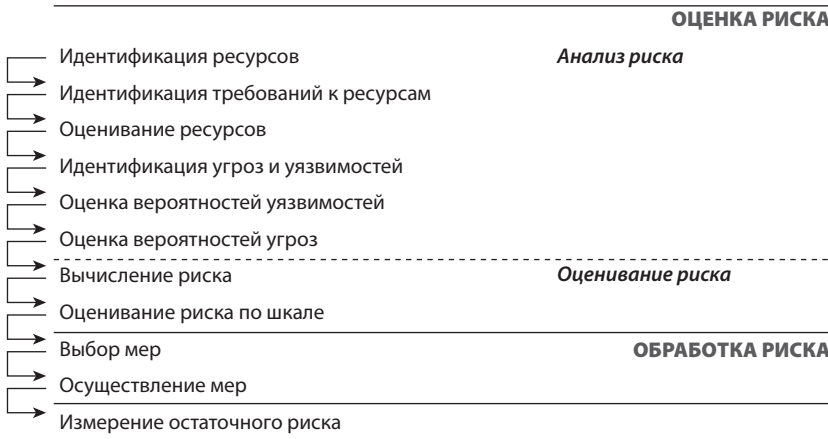

Рис. 1.2. Процедура оценки и обработки риска

После оценки риска принимается решение относительно обработки этого риска – выбору и реализации мер по модификации риска [25, 28, 67, 89, 91]. В основу принятия решения берутся ожидаемые потери, частота возникновения инцидента и другие факторы. В стандарте предложены четыре меры обработки риска:

1. Уменьшение риска, когда риск считается неприемлемым и для его уменьшения выбираются и реализуются соответствующие механизмы безопасности;

2. Передача риска, когда риск считается неприемлемым и на определённых условиях (например, в рамках страхования, поставки или аутсорсинга) переадресуется сторонней организации [15];

3. Принятие риска, если риск считается осознанно допустимым по причине невозможности внедрения разумных мер и средств безопасности.

4. Отказ от риска, когда риск исключается путем отказа от бизнес-процессов организации, являющихся причиной риска.

В результате обработки риска остается так называемый остаточный риск, относительно которого принимается решение о завершении этапа обработки риска.

ГОСТ 27005 не ограничивает использование каких-либо методик и методов анализа риска, как-то: «мозговой штурм», структурированные и полуструктурированные опросы, метод Дельфи, контрольные листы, анализ сценариев, BIA, анализ дерева неисправностей, анализ дерева событий, причинно-следственный анализ, анализ причинно-следственных связей, анализ уровней надежности средств защиты, анализ дерева решений, HRA, анализ диаграммы «галстук-бабочка», цепи Маркова (ТМО), метод Монте-Карло, Байесовский подход и сети Байеса, FN-кривые, метод индексов рисков, матрица последствий и вероятностей, многокритериальный анализ решений и другие [16].

[16] Там же.

2. СЕРТИФИКАЦИЯ СРЕДСТВ ЗАЩИТЫ ИНФОРМАЦИИ

2.1. Определение сертификации средств защиты информации

Под сертификацией понимается подтверждение соответствия третьей стороной, относящееся к продукции, процессам, системам или персоналу[17].

Следует указать ключевые особенности сертификации:

1. Сертификация проводится на соответствие *заданным* требованиям, а именно техническим регламентам [38], положениям стандартов, сводов правил, условиям договоров и другим требованиям, определенным в нормативных документах и соответствующей документации. Поэтому область сертификации и ее результат однозначно определены конкретными нормативными документами, а не требованиями и рекомендациями по повышению качеству или защищенности вообще.

2. В случае положительно результата процесс сертификации заканчивается выдачей официального письменно оформленного удостоверения – сертификата соответствия, а сертифицированная продукция подлежит маркировке знаком соответствия системы сертификации. В некоторых системах сертификации можно встретить еще одно официальное удостоверение – заключение, которое применяется для случаев, когда орган по сертификации затрудняется выдать общепринятый сертификат соответствия.

3. Сертификация является деятельностью третьей стороны, т.е. должна быть обеспечена независимость оценки соответствия, максимально исключающая любые формы аффилированности или сговора [42].

4. Сертификация может быть добровольной и обязательной. Сертификация средств защиты информации по требованиям безопасности информации является обязательной.

---

[17] ГОСТ Р ИСО/МЭК 17000:2009, п.5.5.

5. Так как в стране действует несколько систем сертификации, то эти системы определяют некоторые свои правила и процедуры проведения оценки соответствия, включая аккредитацию органов по сертификации и испытательных лабораторий, разумеется, в рамках российского законодательства и своей компетенции.

Таким образом, сертификация средств защиты информации по требованиям безопасности информации представляет собой обязательное независимое подтверждение соответствия СЗИ требованиям нормативных документов по защите информации с учетом правил федеральных органов (Минобороны, СВР, ФСБ, ФСТЭК) в рамках их компетенции[18]. Следует отметить, что федеральные органы по сертификации трактуют СЗИ в широком смысле, как средство защиты от угроз информационной безопасности и ее составных свойств: целостности, доступности, конфиденциальности и др. В этом смысле под понятие СЗИ при самой общей модели угроз подпадает любое изделие в защищенном исполнении, например, «безопасное» от программных закладок ПО.

В таблице 2.1 приведены примеры объектов сертификации в области информационной безопасности, к которым определены требования в открытых нормативных документах.

Следует прокомментировать табл. 2.1. Например, несмотря на то, что сертификация систем в ФСТЭК проводится в форме аттестации объектов информатизации, последняя реально (при защите конфиденциальной информации) таковой не является, т.к. не полностью соблюдается самый главный закон сертификации о третьей стороне. Несмотря на то, что сформулированы требования к системам менеджмента информационной безопасности (серия ГОСТ 27000), они не нашли отражение в нормативно-методических документах обязательных систем сертификации. В жизни и в документах регуляторов можно встретить классы *общепринятых* СЗИ, таких как: средства контент анализа, средства контроля утечек, средства анализа защищенности, средства управления и мониторинга, средства доверенной загрузки, генераторы паролей, защищенные BIOS, средства безопасности программных приложений, средства безопасности виртуализации, средства безопасности облачных технологий, средства защиты в промышленных системах и системах высокой готовности и др., однако требования к ним либо пока отсутствуют, либо (в противоречие 2-му правилу Керкго́ффса) не подлежат публичному информированию или обсуждению.

[18] См. ФЗ-347.

Таблица 2.1

Объекты сертификации по требованиям безопасности информации

Объекты сертификации	Объект сертификации средств защиты информации
Продукция	Средства защиты информации Средства вычислительной техники Профили защиты Межсетевые экраны Средства обнаружения вторжений Средства антивирусной защиты Средства криптографической защиты информации Средства защиты персональных данных
Системы	Автоматизированные системы
Системы менеджмента	Системы менеджмента (управления) безопасности

Согласно Доктрине информационной безопасности РФ [19] сертификация СЗИ является важнейшим методом обеспечения безопасности страны, а значит, государственную важность приобретает совершенствование мер, направленных на повышение эффективности и достоверности результатов сертификации СЗИ [34]. Именно поэтому процесс сертификации включает несколько уровней независимых проверок: экспертизу заявки в федеральном органе, проведение испытаний в аккредитованной испытательной лаборатории, проверку материалов испытаний в аккредитованном органе по сертификации и др. При этом обеспечивается независимость между участниками сертификации: аккредитованным органом по сертификации, аккредитованной испытательной лабораторией и другими заинтересованными сторонами.

2.2. Правила и участники сертификации средств защиты информации

Согласно Постановлению Правительства РФ 1995 г. № 608, руководство системами сертификации возложено на федеральные органы по сертификации: Минобороны России, СВР России, ФСБ России и ФСТЭК России.

[19] Утв. Президентом РФ 09.09.2000 № Пр-1895.

В общегражданском плане регулирование рынка некриптографических СЗИ в стране возложено на ФСТЭК России, а рынка криптографических СЗИ — на ФСБ России.

Участниками сертификации являются федеральный орган по сертификации, аккредитованный орган по сертификации, аккредитованная испытательная лаборатория, заявитель на сертификацию, которым может быть разработчик, изготовитель или поставщик.

Порядок проведения сертификации выглядит следующим образом [60].

1. Заявитель подает в федеральный орган заявку на проведение сертификационных испытаний.

2. Федеральный орган определяет аккредитованную испытательную лабораторию и орган по сертификации, что фиксируется в решении на сертификацию.

3. Испытательная лаборатория проводит сертификационные испытания.

4. Материалы испытаний (программа и методика, протоколы испытаний, техническое заключение) передаются в орган по сертификации, который проводит их независимую экспертизу.

5. Федеральный орган по сертификации на основании положительного технического заключения органа по сертификации оформляет сертификат соответствия. В случае выявления каких-либо несоответствий федеральный орган может провести дополнительную экспертизу с привлечением экспертов из различных аккредитованных лабораторий и органов по сертификации (см. рис. 2.1).

В области защиты информации применяются следующие схемы сертификации СЗИ:

— сертификация единичного образца СЗИ;

— сертификации партии СЗИ;

— сертификация серии (типового образца) с предварительной проверкой производства.

Сертификационные испытания можно классифицировать по методу тестирования:

— функциональное тестирование продукта или системы по методу «черного ящика»;

— структурное тестирование исходного кода ПО.

В первом случае при испытаниях используются:

— традиционные нормативные документы (например, руководящие документы Гостехкомисии России);

— документация (например, ТУ);

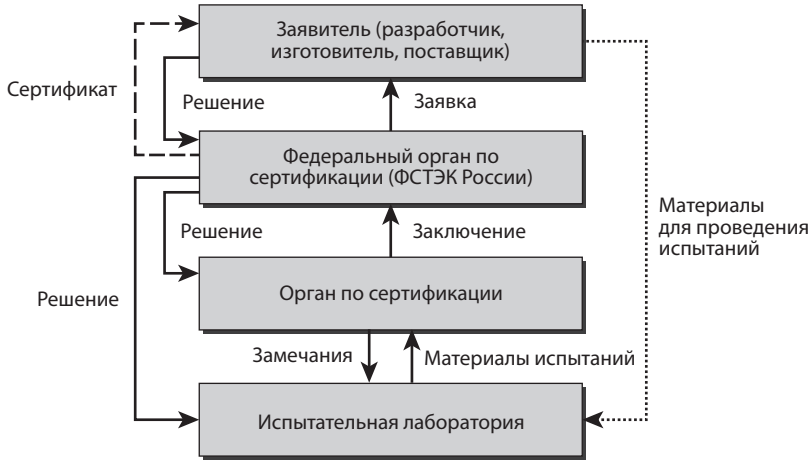

Рис. 2.1. Участники системы сертификации ФСТЭК России

— задание по безопасности — документ, разрабатываемый в соответствии с метастандартом ГОСТ ИСО 15408.

Особенность структурного тестирования состоит в том, что оно проводится в форме статического и динамического анализа исходного кода программ и касается только вопросов внутренней безопасности продукта (контроля отсутствия недекларированных возможностей).

Как ранее отмечалось, документом, подтверждающим положительные результаты сертификационных испытаний, является сертификат соответствия, в котором указаны самые важные моменты: идентификационные характеристики[20], на соответствие каким документам проведены испытания, срок действия, документ (обычно ТУ или формуляр), в котором определены *ограничения* на использование СЗИ и зафиксированы *контрольные суммы* и др.

Перечень аккредитованных испытательных лабораторий ФСТЭК России и ФСБ России, аккредитованных органов по сертификации ФСТЭК, открытые реестры сертифицированных СЗИ, правовые акты и нормативно-методические документы ФСТЭК и ФСБ можно посмотреть на вебпорталах указанных ведомств: www.fstec.ru и clsz.fsb.ru.

Требования к сертификации определены федеральными законами, постановлениями Правительства, стандартами и кодексами,

[20] ГОСТ Р 54011–2010.

а требования к проверкам (сертификационным испытаниям, инженерным или тематическим исследованиям) определены в нормативных документах или в соответствующей документации.

2.3. Законодательно-правовые основы сертификации

Законодательные и правовые требования определяют, когда сертификация необходима, а также ответственность за несоблюдение этих требований.

При определении обязательности сертификации СЗИ удобно провести классификацию защищаемого информационного ресурса и объектов информатизации.

В качестве признаков классификации информационного ресурса выделяют два: принадлежность к государственному информационному ресурсу и уровень ограниченности доступа.

Для государственного информационного ресурса требования устанавливает и контролирует сам собственник (государство). В других случаях могут быть неоднозначности.

При идентификации уровня ограниченности доступа выделяют:

— государственную тайну,

— личную и семейную тайны (персональные данные),

— другие виды тайн, не отнесенные к гостайне и персональным данным,

— открытую общедоступную информацию.

Заметим, что в случае, когда говорят об информации ограниченного доступа, не отнесенной к гостайне, ее исторически часто называют информацией конфиденциального характера или просто конфиденциальной информацией.

Дополнительно могут быть определены требования к системам обработки информации, независимо от классификации обрабатываемой информации.

Назовем основные случаи, когда сертификация СЗИ в нашей стране обязательна (табл. 2.2):

— защищаемая информация составляет сведения, отнесенные к государственной тайне;

— защищаемая информация ограниченного доступа, но не отнесенная к гостайне, при условии, что она относится к государственному информационному ресурсу;

— защищаемая информация относится к персональным данным и составляет личную и семейную тайну;

Таблица 2.2

Основание для требования по сертификации средств защиты информации

Информационный ресурс	Государственная тайна	Личная, семейная тайна	Другие тайны*	Открытая общедоступная информация
Государственный информационный ресурс	Да	Да	Да	Для систем общего пользования и для специфических систем
Негосударственный информационный ресурс	—	Да	Только для специфических систем	Только для специфических систем
* Например: коммерческая, журналистская, депутатская, пенсионная, торговая, промышленная, производственная, профессиональная, служебная, врачебная тайны, ноу-хау, секретный торговый процесс, информация о членах политических партий, инсайдерская информация, а также тайны дневников и личных записей, исповеди, связи, ценных бумаг и еще около 40 разных интересных тайн.				

— к защите объектов информатизации (систем, комплексов) определены требования по оценке соответствия независимо от видов тайн.

К примеру, в случае защиты государственной тайны требования по обязательной сертификации СЗИ определены в Законе РФ «О государственной тайне» 1993 г. № 5485-I, Постановлении Правительства РФ 1995 г. № 608 и в других документах.

Требования по сертификации средств защиты информации конфиденциального характера в государственных организациях определены в Постановлении Правительства РФ 2010 г. № 330 (п.6), а также в нормативных документах ФСБ России и ФСТЭК России.

Требования по сертификации средств защиты персональных данных прямо вытекают из Постановления Правительства РФ 2010 г. № 330 (п.6) и косвенно из Постановления Правительства РФ 2007 г. № 781 (п.5), а также регламентируются нормативными документам ФСБ России и ФСТЭК России (рис. 2.2).

В остальных случаях необходимо руководствоваться нормативными требованиями к специфическим объектам. Примерами таких объектов являются следующие:

— информационные системы критически важных объектов;

> Средства защиты информации, применяемые в информационных системах, в установленном порядке проходят процедуру оценки соответствия.
>
> *Постановление Правительства РФ 2007 г. № 781, п. 5.*

Рис. 2.2. Фрагмент Постановления Правительства РФ № 781

— автоматизированные системы управления технологическим процессом;

— системы управления экологически опасными производствами, объектами, имеющими важное оборонное или экономическое значение и влияющими на безопасность государства;

— федеральные государственные информационные системы общего пользования;

— автоматизированные системы систем вооружений;

— игровые автоматы и др.

Следует сказать, что с практической точки зрения обязательность сертификации СЗИ диктуется обычно двумя обстоятельствами. Первое связано с требованиями заказчика, который формулирует их к разработке, поставке, внедрению защищенной информационной системы. Например, в техническом задании или техническом проекте на ОКР (и дальнейшего авторского надзора или техподдержки) весьма красиво указать ГОСТ Р 51583:2000 «Порядок создания автоматизированных систем в защищенном исполнении». Согласно п. 4.15 этого стандарта необходим сертификат соответствия.

Другой случай связан с необходимостью быть уверенным в защищенности объекта с формальной точки зрения, когда требуется заполучить какой-нибудь официальный документ о подтверждении соответствия информационной системы требованиям российского законодательства. В настоящее время в области информационной безопасности таким документом является аттестат соответствия. Никто не выпишет такой аттестат без сертифицированных СЗИ.

Согласно Закону № 152-ФЗ «О персональных данных» лица, виновные в нарушении требований закона, несут гражданскую, уголовную, административную, дисциплинарную и иную предусмотренную законодательством Российской Федерации ответственность. То же самое можно сказать и про использование несертифицированных СЗИ.

Что касается административных нарушений в области защиты информации, то следует в первую очередь отметить Главу 13 действу-

Таблица 2.3.

Основание для сертификации средств защиты информации

Законодательный или нормативно-правовой документ	Защищаемая информация
Закон «О государственной тайне» 1993 г. № 5485-I	государственная тайна
Постановление Правительства РФ 1995 г. № 608	государственная тайна
Указ Президента РФ 2008 г. № 351	гостайна или служебная тайна (при подключении к информационно-телекоммуникационным сетям международного информационного обмена)
СТР-К	конфиденциальная информация (госучреждений)
Постановление Правительства РФ 2010 г. № 330	информация конфиденциального характера (ГИР); персональные данные
Приказ Председателя Гостехкомиссии России 1995 г. № 199	информация с ограниченным доступом; служебная информация, циркулирующая в системах управления экологически опасными производствами, объектами, имеющими важное оборонное или экономическое значение и влияющими на безопасность государства, а также средствах общего применения, предназначенных для ПДИТР
Указ Президента РФ 1995 г. № 334	криптограммы
Специальные документы ФСТЭК России по ключевым системам информационной инфраструктуры	информационный ресурс информационных систем критически важного объекта
Постановление Правительства РФ 2007 г. № 781	персональные данные
Методические документы 8 Центра ФСБ России по персональным данным	персональные данные
Специальные документы ФСТЭК России по персональным данным	персональные данные
Постановление Правительства РФ 2009 г. № 424	информационный ресурс государственных информационных систем общего пользования
ГОСТ Р 51583–2000	информация ограниченного доступа
ГОСТ Р 51189–1998	служебная тайна

ющего КоАП, в котором весьма интересны для изучения следующие статьи:

— ст. 13.6. Использование несертифицированных средств связи либо предоставление несертифицированных услуг связи;

— ст. 13.12. Нарушение правил защиты информации;

— ст. 13.13. Незаконная деятельность в области защиты информации.

Так, в ст. 13.12 определены административные штрафы для случая использования несертифицированных СЗИ, включая **конфискацию** СЗИ, а при отягчающих обстоятельствах и **высшую меру** административного наказания — приостановление деятельности.

Про УК РФ возникает речь, если внедрение и использование несертифицированных СЗИ квалифицируется как деяние, повлекшее некоторое преступление. Например, согласно ст. 274 УК РФ нарушение правил эксплуатации средств хранения, обработки или передачи охраняемой компьютерной информации, либо информационно-телекоммуникационных сетей и оконечного оборудования, а также правил доступа к информационно-телекоммуникационным сетям, повлекшее уничтожение, блокирование, модификацию либо копирование компьютерной информации может ограничить свободу на *пять* лет. Вопросы нарушения правил и условий, халатности, утраты, разглашения тайн при автоматизированной обработке в той или иной степени отражены в 19, 22, 24, 26—33 главах УК РФ.

Отдельно следует назвать ст. 171 (незаконное предпринимательство), касающуюся деятельности с нарушением обязательных лицензионных требований и условий, в нашем случае, читай, при разработке, внедрении и сертификации СЗИ [87].

Надо понимать, что ответственность за возникшие проблемы в области защиты информации, кроме органов по сертификации и испытательных лабораторий, возлагается также на владельца объекта информатизации, уполномоченного владельцем (по договору) лицо и разработчика.

2.4. Традиционные руководящие документы Гостехкомиссии России

В настоящее время наиболее используемыми в области технической защиты информации (в полном объеме или фрагментарном) во всех системах сертификации являются «традиционные» руководящие документы Гостехкомиссии России [69], разработанные в незапамятные времена про-

шлого века, но остающиеся актуальными до сих пор. Наиболее представительными из них следует назвать следующие четыре:

— РД. Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации (Гостехкомиссия России, 1992 г.);

— РД. Средства вычислительной техники. Межсетевые экраны. Защита от несанкционированного доступа. Показатели защищенности от несанкционированного доступа к информации (Гостехкомиссия России, 1997 г.);

— РД. Автоматизированные системы. Защита от несанкционированного доступа к информации. Классификация автоматизированных систем и требования по защите информации (Гостехкомиссия России, 1992 г.);

— РД. Защита от несанкционированного доступа к информации. Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей (Гостехкомиссия России, 1999 г.).

Первые три документа касаются требований по защите информации, предъявляемых к средствам и системам, и используются при функциональном тестировании (по методу «черного ящика»).

Четвертый документ касается внутренней безопасности программных продуктов (защищенности от уязвимостей) и используется при оценке соответствия структурными методами (по методу «белого ящика»).

2.4.1. Классы защищенности средств вычислительной техники

Руководящий документ «Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации» устанавливает требования к составу документации, а также номенклатуру показателей защищенности средств вычислительной техники (СВТ), описываемых совокупностью требований к защите и определяющих классификацию СВТ по уровню защищенности от НСД к информации [69].

В рамках документа под СВТ понимается совокупность программных и технических элементов систем обработки данных, способных функционировать самостоятельно или в составе других систем, и предназначенных для предотвращения или существенного затруднения несанкционированного доступа к информации. СВТ как

комплексное средство защиты информации от НСД может включать ряд подсистем (механизмов) безопасности, таких как: идентификация, аутентификация, разграничение доступом, контроль целостности, протоколирование и другие механизмы противодействия актуальным угрозам информационной безопасности [1, 11, 45, 46]. В данном РД не предъявляются требования к средствам криптографической защиты информации (СКЗИ), которые, однако, могут быть использованы дополнительно.

Следует заметить, что данный РД релевантен по отношению к стандарту ГОСТ Р 50739–95 «Средства вычислительной техники. Защита от несанкционированного доступа к информации. Общие технические требования».

Документ определяет 7 классов защищенности. Каждый класс характеризуется заданными значениями показателей защищенности СВТ, которые описываются соответствующими требованиями (табл.2.4). Формально требования можно разделить на 4 группы:

– требования к подсистемам идентификации, аутентификации, авторизации (п.п. 1–4, 6–8 в таблице 2.4);

– требования к подсистеме протоколирования (п.п.5, 10);

– требования к гарантиям разработки (п.п.9,11–17);

– требования к документации (п.п.18–21).

Требования к СВТ варьируются по уровню и глубине в зависимости от соответствующего класса защищенности. С точки зрения принципиальных моментов безопасности информации можно выделить три группы СВТ:

– СВТ с гарантированной (верифицированной) защитой информации – класс 1;

– СВТ с полным (мандатным) управлением доступом – классы 2–4;

– СВТ с избирательным (дискретным) управлением доступом – классы 5, 6.

Формально в РД определены 7 классов, но к 7-му классу (в таблице 2.4 не показан) требования не предъявляются, 5-й класс предусмотрен для защиты информации конфиденциального характера, с 4-го по 2-й класс – для защиты сведений, составляющих государственную тайну (соответственно секретных сведений, совершенно секретных, особой важности), 6 и 1 [21] – в настоящее время в РФ не имеют юридической значимости.

[21] Требования по 1 классу предъявляются только к военным системам США (см. DoD 5200.28-STD: 1983).

Таблица 2.4

Показатели защищенности средств вычислительной техники

п/п	Показатель защищенности	Класс защищенности					
		6	5	4	3	2	1*
1	Дискреционный принцип контроля доступа	+	+	+	=	+	=
2	Мандатный принцип контроля доступа	–	–	+	=	=	=
3	Очистка памяти	–	+	+	+	=	=
4	Изоляция модулей	–	–	+	=	+	=
5	Маркировка документов	–	–	+	=	=	=
6	Защита ввода и вывода на отчуждаемый физический носитель информации	–	–	+	=	=	=
7	Сопоставление пользователя с устройством	–	–	+	=	=	=
8	Идентификация и аутентификация	+	=	+	=	=	=
9	Гарантии проектирования	–	+	+	+	+	+
10	Регистрация	–	+	+	+	=	=
11	Взаимодействие пользователя с КСЗ	–	–	–	+	=	=
12	Надежное восстановление	–	–	–	+	=	=
13	Целостность КСЗ	–	+	+	+	=	=
14	Контроль модификации	–	–	–	–	+	=
15	Контроль дистрибуции	–	–	–	–	+	=
16	Гарантии архитектуры	–	–	–	–	–	+
17	Тестирование	+	+	+	+	+	=
18	Руководство для пользователя	+	=	=	=	=	=
19	Руководство по КСЗ	+	+	=	+	+	=
20	Тестовая документация	+	+	+	+	+	=
21	Конструкторская (проектная) документация	+	+	+	+	+	+

* Требования по 1 классу предъявляются только к военным системам США (см. DoD 5200.28-STD: 1983).

Обозначения: «–» — нет требований к данному классу; «+» — новые требования, «=» — требования совпадают с предыдущими; серым фоном выделены требования к защите сведений, составляющих гостайну.

2.4.2. Классы защищенности межсетевых экранов

Документ «РД. Средства вычислительной техники. Межсетевые экраны. Защита от несанкционированного доступа. Показатели защищенности от несанкционированного доступа к информации» разработан для оценки соответствия средств межсетевой защиты (межсетевых экранов), используемых для безопасного разграничения доступа между сегментами сетей.

В РД под межсетевым экраном (МЭ) понимается локальное (однокомпонентное) или функционально-распределенное программное (программно-аппаратное) средство (комплекс), реализующее контроль за информацией, поступающей в автоматизированную систему (АС) и/или выходящей из АС.

Указанным документом определены 12 показателей защищенности МЭ, требования к реализации которых задают класс защищенности МЭ (табл. 2.5).

К каждому классу защищенности МЭ сопоставлено в соответствие требование по защите категории информации ограниченного доступа. Иначе говоря, для того, чтобы АС возможно было аттестовать, объект информатизации должен быть защищен от внешней среды межсетевым экраном не ниже следующего класса:

— 5 класс для АС «1Д», «2Б», «3Б»;

— 4 класс для АС «1Г»;

— 3 класс для АС «1В», а также «2А», «3А» в случае обрабатываемой информации с грифом «секретно»;

— 2 класс для АС «1Б», а также «2А», «3А» в случае обрабатываемой информации с грифом «сов.секретно»;

— 1 класс для АС «1А», а также «2А», «3А» в случае обрабатываемой информации с грифом «особой важности».

2.4.3. Классы защищенности автоматизированных систем

Документ «РД. Автоматизированные системы. Защита от несанкционированного доступа к информации. Классификация автоматизированных систем и требования по защите информации» определяет требования к защищенности информации в АС. Следует сказать, что данный документ может быть основным в случае сертификации системы, но только дополнительным — в случае аттестации. Это связано с тем, что требования по аттестации уточнены специальными нормативными документами ФСТЭК России и национальными стандартами ограниченного доступа.

Таблица 2.5

Показатели защищенности межсетевых экранов

№ п/п	Показатель защищенности	Класс защищенности				
		5	4	3	2	1
1	Управление доступом (фильтрация данных и трансляция адресов)	+	+	+	+	=
2	Идентификация и аутентификация	–	–	+	=	+
3	Регистрация	–	+	+	+	=
4	Администрирование: идентификация и аутентификация	+	=	+	+	+
5	Администрирование: регистрация	+	+	+	=	=
6	Администрирование: простота использования	–	–	+	=	+
7	Целостность	+	=	+	+	+
8	Восстановление	+	=	=	+	+
9	Тестирование	+	+	+	+	+
10	Руководство администратора защиты	+	=	=	=	=
11	Тестовая документация	+	+	+	+	+
12	Конструкторская (проектная) документация	+	=	+	=	+

Документ устанавливает требования к группам подсистем безопасности:

— подсистеме управления доступом (включая идентификацию, аутентификацию и авторизацию);

— подсистеме протоколирования;

— криптографической системе;

— подсистеме обеспечения целостности, а также подсистеме физической защиты, администрирования, тестирования и резервирования.

В РД определены девять классов защищенности АС от несанкционированного доступа к информации (табл. 2.6). Каждый класс задается определенной совокупностью минимальных требований по защите. Классы разбиты на три группы, различающиеся особенностями обработки информации в АС.

Третья группа («3А», «3Б») классифицирует АС, в которых работает один пользователь, допущенный ко всей информации АС, размещенной на носителях одного уровня конфиденциальности.

Вторая группа («2А», «2Б») классифицирует АС, в которых пользователи имеют одинаковые права доступа ко всей информации АС, обрабатываемой на носителях различного уровня конфиденциальности.

Первая группа («1А», «1Б», «1В», «1Г», «1Д») классифицирует многопользовательские АС, в которых одновременно обрабатывается информация разных уровней конфиденциальности и не все пользователи имеют право доступа ко всей информации АС.

В рамках выделенных групп установлено упорядочение требований по защите в зависимости от степени конфиденциальности информации. Класс, имеющий высшую степень защищенности для конкретной группы, отмечается литерой «А» («1А», «2А», «3А»).

Согласно п. 2.18 документа, при разработке АС, предназначенной для обработки или хранения информации, являющейся собственностью государства и отнесенной к категории секретной, необходимо ориентироваться в соответствии с РД «Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации» на классы защищенности АС не ниже (по группам) «3А», «2А», «1А», «1Б», «1В» и использовать сертифицированные СВТ:

— не ниже 4 класса — для класса защищенности АС «1В»;
— не ниже 3 класса — для класса защищенности АС «1Б»;
— не ниже 2 класса — для класса защищенности АС «1А».

Следует обратить внимание на то, что в специальных документах ФСТЭК, используемых при аттестации, требования к п.4.6 таблицы 2.6 однозначно усилены, а к п.3 могут быть снижены за счет ассиметричных мер.

2.4.4. Контроль отсутствия недекларированных возможностей

Документ «РД. Защита от несанкционированного доступа к информации. Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей» определяет требования к структурному анализу ПО с целью выявления недекларированных возможностей (НДВ), под которыми понимаются неописанные в документации функциональные возможности, при использовании которых возможно нарушение уровня безопасности системы. В РД указаны два вида структурного анализа: статический и динамический, что подразумевает необходимость предоставления исходных текстов ПО и спецификаций (в данном случае документации, выполненной в соответствии с ГОСТ).

Таблица 2.6

Классы защищенности автоматизированных систем

№ п/п	Подсистемы и требования	Классы								
		3Б	3А	2Б	2А	1Д	1Г	1В	1Б	1А
1	**Подсистема управления доступом**									
1.1	Идентификация, проверка подлинности и контроль доступа субъектов:									
	в систему	+	+	+	+	+	+	+	+	+
	к терминалам, ЭВМ, узлам сети ЭВМ, каналам связи, внешним устройствам ЭВМ	–	–	–	+	–	+	+	+	+
	к программам	–	–	–	+	–	+	+	+	+
	к томам, каталогам, файлам, записям, полям записей	–	–	–	+	–	+	+	+	+
1.2	Управление потоками информации			–	+	–	–	+	+	+
2	**Подсистема регистрации и учета**									
2.1	Регистрация и учет:									
	входа (выхода) субъектов доступа в (из) систему (ы) (узел сети)	+	+	+	+	+	+	+	+	+
	выдачи печатных (графических) выходных документов	–	+	–	+	–	+	+	+	+
	запуска (завершения) программ и процессов (заданий, задач)	–	–	–	+	–	+	+	+	+
	доступа программ субъектов доступа к защищаемым файлам, включая их создание и удаление, передачу по линиям и каналам связи	–	–	–	+	–	+	+	+	+
	доступа программ субъектов доступа к терминалам, ЭВМ, узлам сети ЭВМ, каналам связи, внешним устройствам ЭВМ, программам, томам, каталогам, файлам, записям, полям записей	–	–	–	+	–	+	+	+	+
	изменения полномочий субъектов доступа	–	–	–	–	–	–	+	+	+

Окончание табл. 2.6

№ п/п	Подсистемы и требования	Классы								
		3Б	3А	2Б	2А	1Д	1Г	1В	1Б	1А
	создаваемых защищаемых объектов доступа	–	–	–	+	–	–	+	+	+
2.2	Учет носителей информации	+	+	+	+	+	+	+	+	+
2.3	Очистка (обнуление, обезличивание) освобождаемых областей оперативной памяти ЭВМ и внешних накопителей	–	+	–	+	–	+	+	+	+
2.4	Сигнализация попыток нарушения защиты	–	–	–	–	–	–	+	+	+
3	**Криптографическая подсистема**									
3.1	Шифрование конфиденциальной информации	–	–	–	+	–	–	–	+	+
3.2	Шифрование информации, принадлежащей различным субъектам доступа (группам субъектов) на разных ключах	–	–	–	–	–	–	–	–	+
3.3	Использование аттестованных (сертифицированных) криптографических средств	–	–	–	+	–	–	–	+	+
4	**Подсистема обеспечения целостности**									
4.1	Обеспечение целостности программных средств и обрабатываемой информации	+	+	+	+	+	+	+	+	+
4.2	Физическая охрана средств вычислительной техники и носителей информации	+	+	+	+	+	+	+	+	+
4.3	Наличие администратора (службы) защиты информации в АС	–	–	–	+	–	–	+	+	+
4.4	Периодическое тестирование СЗИ НСД	+	+	+	+	+	+	+	+	+
4.5	Наличие средств восстановления СЗИ НСД	+	+	+	+	+	+	+	+	+
4.6	Использование сертифицированных средств защиты	–	+	–	+	–	–	+	+	+

Документ определяет четыре уровня контроля отсутствия НДВ, в зависимости от этого предъявляются требования к содержанию проверок, составляющих статический и динамический анализ, а также к составу документации (табл. 2.7). Уровни контроля соответствуют уровню ограниченности доступа к информации, а именно: 4-й уровень контроля соответствует средствам защиты информации конфиденциального характера, с 3-го по 1-й — соответственно средствам защиты секретных сведений, совершенно секретных, особой важности.

Как видно из таблицы 2.7, основное содержание статического анализа составляют процедуры идентификации исходного и загрузочного кода, а также процедуры декомпозиции кода программы вплоть до перечня маршрутов (путей), представляющих собой последовательность выполняемых функциональных объектов. Динамический анализ представляет собой проверку соответствия реальных маршрутов с перечнем маршрутов, полученным на этапе статического анализа.

Таблица 2.7

Уровни контроля отсутствия недекларированных возможностей

№	Наименование требования	Уровень контроля			
		4	3	2	1
Требования к документации					
1	Контроль состава и содержания документации				
1.1	Спецификация (ГОСТ 19.202–78)	+	=	=	=
1.2	Описание программы (ГОСТ 19.402–78)	+	=	=	=
1.3	Описание применения (ГОСТ 19.502–78)	+	=	=	=
1.4	Пояснительная записка (ГОСТ 19.404–79)	-	+	=	=
1.5	Тексты программ, входящих в состав ПО (ГОСТ 19.401–78)	+	=	=	=
Требования к содержанию испытаний					
2	Контроль исходного состояния ПО	+	=	=	=
3	Статический анализ исходных текстов программ				
3.1	Контроль полноты и отсутствия избыточности исходных текстов	+	+	+	=
3.2	Контроль соответствия исходных текстов ПО его объектному (загрузочному) коду	+	=	=	+

Окончание табл. 2.7

№	Наименование требования	Уровень контроля			
		4	3	2	1
3.3	Контроль связей функциональных объектов по управлению	-	+	=	=
3.4	Контроль связей функциональных объектов по информации	-	+	=	=
3.5	Контроль информационных объектов	-	+	=	=
3.6	Контроль наличия заданных конструкций в исходных текстах	-	-	+	+
3.7	Формирование перечня маршрутов выполнения функциональных объектов	-	+	+	=
3.8	Анализ критических маршрутов выполнения функциональных объектов	-	-	+	=
3.9	Анализ алгоритма работы функциональных объектов на основе блок-схем, диаграмм и т. п., построенных по исходным текстам контролируемого ПО	-	-	+	=
4	Динамический анализ исходных текстов программ				
4.1	Контроль выполнения функциональных объектов	-	+	+	=
4.2	Сопоставление фактических маршрутов выполнения функциональных объектов и маршрутов, построенных в процессе проведения статического анализа	-	+	+	=
5	Отчетность	+	+	+	+

Следует обратить внимание на контроль по 2-му уровню, так как здесь предусмотрены проверки по безопасности кода, а именно контроль конструкций (предполагается, что это фрагменты потенциально опасного кода) и анализ критических (потенциально небезопасных) маршрутов.

2.5. Требования к защите персональных данных

В настоящее время к категории конфиденциальной информации, имеющей характер персональных данных (составляющих личную и семейную тайну), предъявляются особые нормативно-методи-

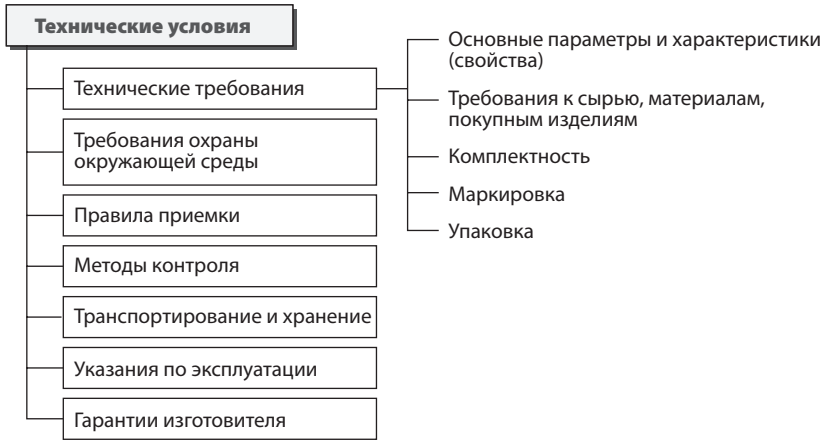

Рис. 2.3. Структура ТУ по ГОСТ 2.114–95

ческие требования к оценке соответствия их средств защиты. При оценке соответствия выделяют три момента:

– классификация информационных систем персональных данных (ИСПДн);

– определение требований к составу и классам защищённости средств защиты;

– формирование требований к СЗИ в конструкторском документе «Технические условия» (главным образом, в разделах «технические требования» и «указания по эксплуатации»), в соответствии с которыми и проводится оценка соответствия (рис. 2.3).

Правила классификации введены Приказом ФСТЭК России, ФСБ России, Мининформсвязи России 2009 г. № 55/86/20 «Об утверждении порядка проведения классификации информационных систем персональных данных». Согласно правилам, класс ИСПДн зависит от категории обрабатываемых персональных данных и их объема (табл. 2.8).

В нормативно-методических документах регуляторов указаны следующие средства защиты информации:

– средства предотвращения несанкционированного доступа;

– средства защиты информации при межсетевом взаимодействии;

– антивирусные средства;

– средства анализа защищенности;

– средства обнаружения вторжений;

– криптографические средства.

Таблица 2.8

Классификация информационных систем персональных данных

Категории персональных данных	Число субъектов персональных данных		
	До 1000 (организация)	1000—100 000 (отрасль, город)	Более 100 000 (субъект федерации)
Обезличенные, общедоступные данные	ИСПДн К4	ИСПДн К4	ИСПДн К4
Идентификационные данные	ИСПДн К3	ИСПДн К3	ИСПДн К2
Дополнительные данные	ИСПДн К3	ИСПДн К2	ИСПДн К1
Данные о состоянии здоровья, расовой и национальной принадлежности, политических, религиозных и философских взглядах, интимной жизни	ИСПДн К1	ИСПДн К1	ИСПДн К1

Таблица 2.9

Требования к системам и средствам защиты персональных данных

Класс ИСПДн	Класс АС	Класс МЭ	Класс МЭ при защите СОП	Уровень контроля НДВ	Уровень доверия к СОВ	Уровень доверия к антивирусному средству
ИСПДн К1	3А, 2А-, 1Г- **	5	3	4	ОУД 3+	ОУД 3+
ИСПДн К2	3Б, 2Б, 1Д	5	4	*	ОУД 2+	ОУД 2+
ИСПДн К3	3Б, 2Б, 1Д	5	5	*	ОУД 1+	ОУД 1+
ИСПДн К4	*	*	*	*	ОУД 1+	ОУД 1+
** Знак «-» означает, что требования для указанного класса заданы не в полном объеме; знак «+» — требования усилены; * — означает, что требования определяются оператором.						

Независимо от класса ИСПДн, средства криптографической защиты информации (СКЗИ) имеют легитимность только при сертификации по требованиям ФСБ России. Уровень криптографической защиты персональных данных, обеспечиваемой СКЗИ, определяется оператором путем отнесения нарушителя (действиям которого должно противостоять криптосредство) к конкретному типу.

К некриптографическим СЗИ требования определены Приказом ФСТЭК России 2010 г. № 58, а также документами ФСТЭК России, касающимся систем обнаружения вторжений и систем антивирусной защиты. С учетом традиционных руководящих документов Гостехкомиссии России легко получить сводную таблицу требований к подсистемам защиты ИСПДн (табл. 2.9) [57].

Из таблицы 2.9 видно, что в настоящее время только по средствам анализа защищенности пока еще не сформированы требования по безопасности информации.

2.6. Требования к защите информационных систем общего пользования

Несмотря на то, что информационный ресурс федеральных информационных систем общего пользования (порталов, оказывающих конституционные услуги) открытый и *общедоступный*, он подлежит защите сертифицированными средствами. Например, Постановлением Правительства РФ 2009 г. № 424 «Об особенностях подключения федеральных государственных информационных систем к информационно-телекоммуникационным сетям» определено, что при подключении таких систем к информационно-телекоммуникационным сетям, доступ к которым не ограничен (например, метасеть Интернет), необходимо использовать СЗИ, прошедшие оценку соответствия.

Дополнительные требования к СЗИ определены совместным Приказом ФСБ России и ФСТЭК России 2010 г. № 416/№ 484 «Об утверждении требований о защите информации, содержащихся в информационных системах общего пользования» в зависимости от класса системы. Приказ обозначил два класса информационных систем общего пользования (ИСОП):

— системы 1-го класса, в которому относятся ИСОП Правительства РФ и другие ИСОП, нарушение целостности и доступности информации, содержащейся в них, может привести к возникновению угроз безопасности страны.

Таблица 2.10

Средства защиты информации систем общего пользования

Классы средств защиты информации	Сертификат соответствия на СЗИ	
	ИСОП 1	ИСОП 2
СКЗИ (ЭЦП)	ФСБ	ФСБ
СЗИ от неправомерных действий	ФСБ	ФСБ или ФСТЭК
Антивирусные средства	ФСБ	ФСБ или ФСТЭК
Средства контроля доступа	ФСБ	ФСБ или ФСТЭК
Системы обнаружения компьютерных атак	ФСБ	ФСБ или ФСТЭК
Межсетевые экраны	ФСБ	ФСБ или ФСТЭК

— системы 2-го класса, к которым относятся остальные все ИСОП.

Решение об отнесении к классу ИСОП определяется руководителем соответствующего федерального органа исполнительной власти.

Особенностью объединенного Приказа является введение требований по использованию сертифицированных классов СЗИ, представленных в табл. 2.10.

Заметим, что конкретные требования к классам защищенности средств защиты общедоступной информации не определены в открытой печати, за исключением систем обнаружения вторжений и антивирусных средств для ИСОП 2. Согласно нормативным документам ФСТЭК России названные средства должны соответствовать ОУД 2+ (усиленный). В остальных случаях необходимо ориентироваться на модель нарушителя, модель угроз и технологии обрабатываемой информации.

2.7. Общие критерии оценки безопасности информационных технологий

В настоящее время в различных системах сертификации проводятся изыскания по созданию и внедрению международной нормативной базы оценки соответствия на основе стандарта ГОСТ Р ИСО/МЭК 15408 «Информационная технология. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий». Особенность стандарта состоит в том, что он

является фактически метастандартом, позволяющим создавать нормативные документы к ИТ-продуктам, причем включающим конкретные требования как по безопасности (функциональные требования), так и по качеству (требования доверия), и которые можно представить в полуформализованном виде [55, 80, 84].

Следует отметить, что стандарту в редакции 2000-го года[22] практически идентичен руководящий документ «Безопасность информационных технологий. Критерии оценки безопасности информационных технологий. Части 1, 2, 3» (Гостехкомиссия России, 2002 г.). Именно этот документ является основополагающим при организации сертификации по инновационным правилам. За этими документами исторически закрепилось название «Общие критерии» (ОК), и мы будем придерживаться этой терминологии.

Следует назвать еще нормативно-методические документы ФСТЭК России, разработанные по линии ОК, например:

— Безопасность информационных технологий. Положение по разработке профилей защиты и заданий по безопасности (Гостехкомиссия России, 2003 г.);

— Безопасность информационных технологий. Руководство по регистрации профилей защиты (Гостехкомиссия России, 2003 г.);

— Безопасность информационных технологий. Руководство по формированию семейств профилей защиты (Гостехкомиссия России, 2003 г.);

— Руководство по разработке профилей защиты и заданий по безопасности (Гостехкомиссия России, 2003);

— Требования к системам обнаружения вторжений (ФСТЭК России, 2011 г.);

— Требования к средствам антивирусной защиты (ФСТЭК России, 2012 г.);

— пакет методических документов по профилям защиты [69].

2.7.1. Модель критериев оценки безопасности информационных технологий

«Общие критерии» включают три части:

1. Введение и общая модель;
2. Функциональные требования безопасности;
3. Требования доверия к безопасности.

---

[22] В н.вр. ожидается редакция ГОСТ 15408—2012, аутентичная ISO/IEC 15408:2008,2009 и Common Criteria 3.1.

В первой части нормативного документа даются концептуальные определения процесса оценки соответствия по ОК. В частности, объект испытаний – объект оценки (ОО) рассматривается не сам по себе, а в контексте окружающей среды. При подготовке к оценке соответствия предполагается, что должны быть формализованы следующие так называемые *аспекты среды ОО*:

– предположения безопасности;
– угрозы безопасности;
– политики безопасности.

Предположения безопасности выделяют ОО из общего контекста и задают границы его рассмотрения. Предполагается, что среда ОО удовлетворяет данным предположениям. При проведении оценки предположения безопасности принимаются без доказательств.

Угрозы безопасности характеризуются следующими параметрами:

– источник угрозы;
– предполагаемый способ реализации угрозы;
– уязвимости, которые являются предпосылкой для реализации угрозы;
– активы, которые являются целью нападения;
– нарушаемые свойства безопасности активов;
– возможные последствия реализации угрозы.

Выделяются только те угрозы, наличие которых в рассматриваемой среде установлено или предполагается.

Аналогично, излагаются положения *политики безопасности*, применяемые в организации, которые имеют непосредственное отношение к ОО.

На основании сформулированных предположений безопасности, при учёте угроз и политик формулируются *цели безопасности* для ОО, направленные на обеспечение противостояния угрозам и выполнение положений политики безопасности.

Для достижения поставленных целей к ОО и его среде предъявляются *требования безопасности*.

Именно 2-ая и 3-я части нормативного документа представляют собой каталоги требований безопасности следующих типов:

– функциональные требования – предъявляются к функциям безопасности ОО и реализующим их механизмам безопасности.

– требования доверия – предъявляются к технологии и процессу разработки, эксплуатации и оценки ОО и призваны гарантировать адекватность реализации механизмов безопасности.

При формулировании требований к ОО могут быть разработаны два документа:

– профиль защиты (ПЗ);
– задание по безопасности (ЗБ).

Профиль защиты представляет собой не зависящую от конкретной реализации совокупность требований для некоторой категории ОО. Он не привязан к конкретному ОО и представляет собой обобщённый стандартный набор функциональных требований и требований доверия для определённого класса продуктов (рис. 2.4). Именно каталог утверждённых (сертифицированных) ПЗ, думается, должен послужить заменой традиционных руководящих документов Гостехкомиссии России.

Задание по безопасности – это документ, содержащий требования безопасности для конкретного ОО и специфицирующий функции безопасности и меры доверия, предлагаемые ОО для выполнения установленных требований (рис. 2.5). В ЗБ может быть заявлено соответствие одному или нескольким ПЗ.

Рис. 2.4. Структура профиля защиты

Рис. 2.5. Структура задания по безопасности

В некотором смысле ЗБ можно рассматривать как техническое задание на подсистему обеспечения информационной безопасности для ОО. Именно ЗБ служит основой для проведения оценки ОО с целью демонстрации соответствия его требованиям безопасности.

2.7.2. Функциональные требования безопасности

Систематизированный каталог функциональных требований безопасности сосредоточен во 2-й части ОК. В текущей редакции стандарта функциональные требования разбиты на 11 классов, 66 се-

Рис. 2.6. Структура функционального класса

мейств и 135 компонентов. Структура функционального класса приведена на рис. 2.6.

В таблице 2.11 приведена краткая характеристика всех 66 семейств функциональных требований.

Наличие каталога функциональных требований не предполагает их окончательный и всеобъемлющий характер. В случае необходимо-

Таблица 2.11

Семейства функциональных требований

№ п/п	Семейство	Наименование	Характеристика
Класс *FAU— аудит безопасности*			
1	FAU_ARP	Автоматическая реакция аудита безопасности	Определяются действия, которые необходимо предпринять при выявлении возможных нарушений безопасности.
2	FAU_GEN	Генерация данных аудита	Выбираются события, потенциально подвергаемые аудиту и протоколированию. Определяется минимум регистрируемых данных о событиях безопасности. Осуществляется привязка событий к идентификаторам вызвавших их пользователей.

Продолжение табл. 2.11

№ п/п	Семейство	Наименование	Характеристика
3	FAU_SAA	Анализ аудита безопасности	Устанавливаются требования к средствам аудита безопасности, функционирующим: — на базе правил; — на базе профилей поведения пользователей; — на базе сигнатур атак.
4	FAU_SAR	Просмотр аудита безопасности	Определяются требования к представлению данных аудита. Предоставляются права на просмотр записей аудита уполномоченным пользователям.
5	FAU_SEL	Выбор событий аудита безопасности	Определяются требования по отбору реально протоколируемых событий из числа потенциально подвергаемых протоколированию. Выделяются атрибуты, по которым производится отбор событий.
6	FAU_STG	Хранение данных аудита безопасности	Определяются требования по защите данных аудита от НСД и повреждения. Определяется последовательность действий, выполняемых системой при переполнении журнала аудита.
Класс *FCO— связь*			
7	FCO_NRO	Неотказуемость отправления	Задаются требования по ассоциации атрибутов отправителя информации с элементами передаваемых данных.
8	FCO_NRR	Неотказуемость получения	Задаются требования по ассоциации атрибутов получателя информации с элементами передаваемых данных.
Класс *FCS— криптографическая поддержка*			
9	FCS_CKM	Управление криптографическими ключами	Задаются требования к реализации механизмов генерации, распределения и уничтожения криптографических ключей.
10	FCS_COP	Криптографические операции	Декларируется использование тех или иных криптографических средств защиты информации.

Продолжение табл. 2.11

№ п/п	Семейство	Наименование	Характеристика
Класс *FDP— защита данных пользователя*			
11	FDP_ACC	Политика управления доступом	Идентифицируются применяемые политики управления доступом.
12	FDP_ACF	Функции управления доступом	Описываются правила работы функций, реализующих заявленные политики безопасности.
13	FDP_DAU	Аутентификация данных	Определяется порядок генерации и использования данных аутентификации.
14	FDP_ETC	Экспорт данных за пределы действия ФБО	Определяется порядок экспорта данных за пределы области действия функций безопасности объекта оценки.
15	FDP_IFC	Политика управления информационными потоками	Устанавливаются имена и определяются области действия политик, управляющих информационными потоками. Задаются требования к покрытию данными политиками всех операций перемещения информации.
16	FDP_IFF	Функции управления информационными потоками	Требуется наличие правил управления информационными потоками. Определяются правила контроля информационных потоков и управления ими.
17	FDP_ITC	Импорт данных из-за пределов действия ФБО	Определяется порядок импорта данных из-за пределов области действия функций безопасности объекта оценки.
18	FDP_ITT	Передача в пределах ОО	Определяется порядок защиты данных пользователя при их передаче между различными частями ОО по внутреннему каналу.
19	FDP_RIP	Защита остаточной информации	Задаются требования по уничтожению предыдущего содержания ресурсов АС при их освобождении.
20	FDP_ROL	Откат	Требуется обеспечение возможности отмены последней выполненной операции (или ряда операций) и возврата к предыдущему состоянию.

Продолжение табл. 2.11

№ п/п	Семейство	Наименование	Характеристика
21	FDP_SDI	Целостность хранимых данных	Задаются требования по контролю целостности данных пользователя.
22	FDP_UCT	Защита конфиденциальности данных пользователя при передаче между ФБО	Определяются требования по обеспечению конфиденциальности данных пользователя при их передаче по внешнему каналу между ОО и доверенными внешними объектами ИТ.
23	FDP_UIT	Защита целостности данных пользователя при передаче между ФБО	Определяются требования по защите целостности данных пользователя при передаче между ФБО. Задаются требования к механизмам коррекции ошибок.
Класс *FIA — идентификация и аутентификация*			
24	FIA_AFL	Отказы аутентификации	Задаётся реакция на неудачные запросы аутентификации.
25	FIA_ATD	Определение атрибутов пользователя	Определяются атрибуты пользователей, отличные от идентификаторов и используемые для реализации установленных политик.
26	FIA_SOS	Спецификация секретов	Задаются требования к механизмам проверки качества и генерации секретов (данных аутентификации).
27	FIA_UAU	Аутентификация пользователя	Задаются требования к реализации механизмов аутентификации. Определяются механизмы, доступные до осуществления аутентификации пользователя.
28	FIA_UID	Идентификация пользователя	Задаётся порядок идентификации пользователя. Определяются действия, которые могут быть выполнены до идентификации пользователя.
29	FIA_USB	Связывание пользователь-субъект	Определяется связь атрибутов безопасности пользователя с субъектом, действующим от имени пользователя.

Продолжение табл. 2.11

№ п/п	Семейство	Наименование	Характеристика
Класс *FMT — управление безопасностью*			
30	FMT_MOF	Управление отдельными функциями ФБО	Определяются пользователи, уполномоченные осуществлять управление режимами выполнения функций безопасности.
31	FMT_MSA	Управление атрибутами безопасности	Определяются пользователи, уполномоченные осуществлять управление атрибутами безопасности. Регламентируется порядок контроля безопасности параметров данных атрибутов.
32	F M T _ MTD	Управление данными ФБО	Определяются пользователи, уполномоченные осуществлять управление ФБО. Определяются граничные значения данных функций безопасности и действия в случае выхода за допустимые границы.
33	FMT_REV	Отмена	Определяется порядок отмены атрибутов безопасности пользователей, субъектов и объектов.
34	FMT_SAE	Срок действия атрибута безопасности	Задаются мероприятия, выполняемые по окончании срока действия атрибутов безопасности.
35	FMT_SMR	Роли управления безопасностью	Задаются различные роли пользователей системы и создаются правила, управляющие отношениями между ролями.
Класс *FPR — приватность*			
36	FPR_ANO	Анонимность	Задаётся возможность выполнения определённых действий без запроса идентификатора пользователя
37	FPR_PSE	Псевдонимность	Определяется возможность использования ресурсов без раскрытия идентификатора пользователя, но с сохранением подотчётности.
38	FPR_UNL	Невозможность ассоциации	Требуется невозможность ассоциирования пользователя с применяемым им сервисом (может потребоваться для защиты от построения поведенческих моделей пользователя).

Продолжение табл. 2.11

№ п/п	Семейство	Наименование	Характеристика
39	FPR_UNO	Скрытность	Требуется предоставление пользователю возможности работы с определёнными сервисами незаметно для кого бы то ни было.
Класс *FPT — защита ФБО*			
40	FPT_AMT	Тестирование базовой абстрактной машины	Задаются требования к тестированию, демонстрирующему правильность предположений, обеспечиваемых программно-аппаратной платформой, лежащей в основе функций безопасности.
41	FPT_FLS	Безопасность при сбое	Перечисляются типы сбоев, которые не должны приводить к нарушению безопасности системы.
42	FPT_ITA	Доступность экспортируемых данных ФБО	Определяются правила предотвращения потери доступности экспортируемых данных функций безопасности.
43	FPT_ITC	Конфиденциальность экспортируемых данных ФБО	Определяются правила защиты от несанкционированного раскрытия экспортируемых данных функций безопасности.
44	FPT_ITI	Целостность экспортируемых данных ФБО	Определяются правила защиты от несанкционированной модификации экспортируемых данных функций безопасности.
45	FPT_ITT	Передача данных ФБО в пределах ОО	Регламентируются требования защиты данных функций безопасности при их передаче между разделенными частями ОО по внутреннему каналу
46	FPT_PHP	Физическая защита ФБО	Требуется наличие средств выявления и реагирования на несанкционированный физический доступ к компонентам ОО.
47	FPT_RCV	Надежное восстановление	Требуется наличие возможности корректного автоматического или ручного восстановления функций безопасности после сбоев.

Продолжение табл. 2.11

№ п/п	Семейство	Наименование	Характеристика
48	FPT_RPL	Обнаружение повторного использования	Задаются требования по обнаружению повторного использования сущностей различных типов и последующими действиями по его устранению.
49	FPT_RVM	Посредничество при обращениях	Задаются требования по реализации концепции монитора безопасности обращений.
50	FPT_SEP	Разделение домена	Формулируются требования по организации защищённого домена для каждой функции безопасности.
51	FPT_SSP	Протокол синхронизации состояний	Требуется надёжное подтверждение при обмене данными между функциями безопасности в распределённой среде.
52	FPT_STM	Метки времени	Требуется предоставление надёжных меток времени в пределах ОО (что необходимо, например, для корректной работы механизмов протоколирования).
53	FPT_TDC	Согласованность данных ФБО между ФБО	Задаются требования по согласованности интерпретации данных, совместно используемых различными функциями безопасности и другими доверенными изделиями ИТ.
54	FPT_TRC	Согласованность данных ФБО при дублировании в пределах ОО	Определяются требования по синхронизации данных, дублируемых в пределах ОО.
55	FPT_TST	Самотестирование ФБО	Задаются требования по самотестированию функций безопасности в части типичных операций с известным результатом.
Класс *FRU— использование ресурсов*			
56	FRU_FLT	Отказоустойчивость	Требуется корректное выполнение части функциональных возможностей в случае сбоев.
57	FRU_PRS	Приоритет обслуживания	Определяется порядок применения высокоприоритетных операций.

Окончание табл. 2.11

№ п/п	Семейство	Наименование	Характеристика
58	FRU_RSA	Распределение ресурсов	Задаются требования к механизму квотирования, используемому для достижения высокой доступности ресурсов.
Класс *FTA — доступ к ОО*			
59	FTA_LSA	Ограничение области выбираемых атрибутов	Определяются ограничения на атрибуты безопасности сеанса, которые может выбирать пользователь.
60	FTA_MCS	Ограничение на параллельные сеансы	Задаются требования по ограничению числа параллельных сеансов, предоставляемых одному и тому же пользователю.
61	FTA_SSL	Блокирование сеанса	Определяется возможность блокирования и разблокирования интерактивного сеанса работы пользователя (по желанию пользователя или по инициативе функций безопасности).
62	FTA_TAB	Предупреждения перед предоставлением доступа к ОО	Определяются требования к отображению для пользователей предупреждающего сообщения относительно характера использования ОО.
63	FTA_TAH	История доступа к ОО	Задаются требования по отображению для пользователя при успешном открытии сеанса истории успешных и неуспешных попыток получить доступ от имени этого пользователя.
64	FTA_TSE	Открытие сеанса с ОО	Задаются параметры функций безопасности, на основании которых пользователю может быть отказано в доступе.
Класс *FTP — доверенный маршрут/канал*			
65	FTP_ITC	Доверенный канал передачи между ФБО	Определяются правила организации доверенного канала между функциями безопасности и другими доверенными продуктами ИТ. Определяется порядок идентификации взаимодействующих сторон.
66	FTP_TRP	Доверенный маршрут	Определяется порядок организации канала защищённого взаимодействия между пользователями и функциями безопасности.

сти, функциональные требования безопасности, которые отсутствуют в каталоге, могут быть сформулированы в явном виде[23].

2.7.3. Требования доверия к безопасности

Требования доверия, приведённые в 3-ей части текущей редакции ОК, сгруппированы в 10 классов, 44 семейства и 93 компонента.

Доверие — степень для уверенности в том, что продукт или система ИТ отвечают целям безопасности[24]. ОК обеспечивают доверие с использованием различных методов оценки, например, таким как:

— анализ и проверка процессов и процедур;

— проверка, что процессы и процедуры действительно применяются;

— анализ соответствия между представлениями проекта ОО;

— анализ соответствия каждого представления проекта ОО требованиям;

— верификация доказательств;

— анализ руководств;

— анализ разработанных функциональных тестов и полученных результатов;

— независимое функциональное тестирование;

— анализ уязвимостей, включающий предположения о недостатках;

— тестирование проникновением.

Наиболее общую совокупность требований доверия называют *классом*. Каждый класс содержит семейства доверия, которые разделены на компоненты доверия, содержащие, в свою очередь, элементы доверия.

Структура класса доверия приведена на рис. 2.7.

Все семейства доверия в части 3 ОК являются линейно иерархическими, хотя линейность не обязательна для семейств доверия, которые могут быть добавлены в дальнейшем. В таблице 2.12 приведена краткая характеристика всех 44 семейств доверия.

На практике при разработке ПЗ и ЗБ рекомендуется оформлять требования доверия к ОО в виде одного из определённых в Части 3 ОК оценочных уровней доверия.

Оценочный уровень доверия (ОУД) представляет собой рассчитанную на многократное применение комбинацию требований доверия,

---

[23] ГОСТ Р ИСО/МЭК 15408–2-2008.

[24] ГОСТ Р ИСО/МЭК 15408–3-2008.

Рис. 2.7. Структура класса доверия

Таблица 2.12

Семейства доверия

№ п/п	Семейство	Наименование	Характеристика
Класс *ACM — управление конфигурацией* (УК)			
1	ACM_AUT	Автоматизация УК	Устанавливается уровень автоматизации, используемый для управления элементами конфигурации
2	ACM_CAP	Возможности УК	Определяются функциональные характеристики системы управления конфигурацией

Продолжение табл. 2.12

№ п/п	Семейство	Наименование	Характеристика
3	ACM_SCP	Область УК	Указываются те элементы ОО, для которых необходим контроль со стороны системы управления конфигурацией.
Класс *ADO— поставка и эксплуатация*			
4	ADO _DEL	Поставка	Задаются процедуры, используемые для поддержки безопасности во время передачи ОО пользователю при первоначальной поставке и при последующих модификациях.
5	ADO _IGS	Установка, генерация и запуск	Обеспечивается, чтобы копия ОО была конфигурирована и активизирована администратором так, чтобы показать те же самые свойства защиты, что и у оригинала ОО.
Класс *ADV— разработка*			
6	ADV _FSP	Функциональная спецификация	Предъявляются требования к составу и содержанию *функциональной спецификации*, описывающей функции безопасности ОО.
7	ADV _HLD	Проект верхнего уровня	Предъявляются требования к составу и содержанию *проекта верхнего уровня* — проектной спецификации самого высокого уровня, которая уточняет функциональную спецификацию ФБО в основных составляющих частях ФБО.
8	ADV _IMP	Представление реализации	Предъявляются требования к *представлению реализации* — наименее абстрактному представлению ФБО. Оно фиксирует детализированное внутреннее содержание ФБО на уровне исходного текста, аппаратных схем и т.д.
9	ADV _INT	Внутренняя структура ФБО	Задаётся порядок внутреннего структурирования функций безопасности ОО.

Продолжение табл. 2.12

№ п/п	Семейство	Наименование	Характеристика
10	ADV _LLD	Проект нижнего уровня	Задаются требования к составу и содержанию *проекта нижнего уровня* — детализированной проектной спецификации, уточняющей проект верхнего уровня до уровня детализации, который может быть использован как основа для программирования и/или проектирования аппаратуры.
11	ADV _RCR	Соответствие представлений	Требуется демонстрация отображения между всеми смежными парами имеющихся представлений ФБО, от краткой спецификации ОО до наименее абстрактного из имеющихся представлений ФБО.
12	ADV _SPM	Моделирование политики безопасности	Требуется необходимость использования *моделей политики безопасности* — структурных представлений политик безопасности ПБО, используемых для обеспечения повышенного доверия тому, что функциональная спецификация соответствует политикам безопасности из ПБО.
Класс *AGD— руководства*			
13	AGD_ADM	Руководство администратора	Задаются требования к составу и содержанию руководства администратора.
14	AGD_USR	Руководство пользователя	Задаются требования к составу и содержанию руководства пользователя.
Класс *ALC— поддержка жизненного цикла*			
15	ALC_DVS	Безопасность разработки	Определяются физические, процедурные, относящиеся к персоналу и другие меры безопасности, используемые применительно к среде разработки.

Продолжение табл. 2.12

№ п/п	Семейство	Наименование	Характеристика
16	ALC_FLR	Устранение недостатков	Требуется, чтобы недостатки, обнаруженные потребителями ОО, отслеживались и исправлялись в ходе сопровождения ОО разработчиком. Оцениваются политики и процедуры, которые разработчик предусмотрел для выявления и устранения недостатков и распространения исправлений потребителям.
17	ALC_LCD	Определение жизненного цикла	Задаются требования к технологии разработки, используемой разработчиком для создания ОО.
18	ALC_TAT	Инструментальные средства и методы	Задаются требования к инструментальным средствам разработки, используемым для анализа и создания ОО.
Класс *ATE — тестирование*			
19	ATE _COV	Покрытие	Предъявляются требования к анализу полноты функциональных тестов, выполненных разработчиком для ОО.
20	ATE _DPT	Глубина	Определяется уровень детализации, на котором разработчик проверяет ОО.
21	ATE _FUN	Функциональное тестирование	Задаются требования к содержанию функционального тестирования, выполняемого разработчиком.
22	ATE _IND	Независимое тестирование	Определяется объём и порядок независимого контроля результатов функционального тестирования.
Класс *AVA — оценка уязвимостей*			
23	AVA_CCA	Анализ скрытых каналов	Определяется порядок выявления скрытых каналов передачи информации.

Продолжение табл. 2.12

№ п/п	Семейство	Наименование	Характеристика
24	AVA_MSU	Неправильное применение	Определяется порядок анализа способности администратора или пользователя, используя руководства, определить, что ОО конфигурирован или эксплуатируется небезопасным способом.
25	AVA_SOF	Стойкость функций безопасности ОО	Определяется порядок анализа стойкости функций безопасности ОО, которые реализованы с помощью вероятностного или перестановочного механизма (например, пароля или хэш-функции).
26	AVA_VLA	Анализ уязвимостей	Определяется порядок анализа недостатков, которые могли быть внесены на различных этапах разработки.
Класс *AMA — поддержка доверия*			
27	AMA_AMP	План поддержки доверия	Идентифицируются планы и процедуры, которые выполняются разработчиком для обеспечения поддержки доверия, установленного к оцененному ОО, после изменений в ОО или его среде.
28	AMA_CAT	Отчет о категорировании компонентов ОО	Определяется порядок категорирования компонентов ОО (например, подсистем ФБО) по их отношению к безопасности.
29	AMA_EVD	Свидетельство поддержки доверия	Определяется порядок поддержки разработчиком доверия к ОО в соответствии с планом поддержки доверия.
30	AMA_SIA	Анализ влияния на безопасность	Задаётся порядок проводимого разработчиком анализа влияния на безопасность всех изменений, воздействующих на ОО после его оценки.

Окончание табл. 2.12

№ п/п	Семейство	Наименование	Характеристика
Класс *APE — оценка профиля защиты*			
31	APE_DES	Описание ОО	Определяется порядок контроля состава и содержания соответствующих разделов профиля защиты.
32	APE_ENV	Среда безопасности	
33	APE_INT	Введение ПЗ	
34	APE_OBJ	Цели безопасности	
35	APE_REQ	Требования безопасности ИТ	
36	APE_SRE	Требования безопасности ИТ, сформулированные в явном виде	
Класс *ASE — оценка задания по безопасности*			
37	ASE_DES	Описание ОО	Определяется порядок контроля состава и содержания соответствующих разделов задания по безопасности.
38	ASE_ENV	Среда безопасности	
39	ASE_INT	Введение ЗБ	
40	ASE_OBJ	Цели безопасности	
41	ASE_PPC	Утверждения о соответствии ПЗ	
42	ASE_REQ	Требования безопасности ИТ	
43	ASE_SRE	Требования безопасности ИТ, сформулированные в явном виде	
44	ASE_TSS	Краткая спецификация ОО	

содержащую не более одного компонента из каждого семейства доверия.

В стандарте определены 7 оценочных уровней доверия. С возрастанием порядкового номера предъявляемые требования усиливаются.

ОУД 1 предусматривает *функциональное тестирование*. ОУД 1 применим в тех случаях, когда требуется некоторая уверенность в правильном функционировании ОО, а угрозы безопасности не рассматриваются как серьезные. Он может быть полезен там, где требуется независимое подтверждение утверждения о том, что было уделено должное внимание защите персональных данных или подобной информации. Предполагается, что оценка ОУД 1 может успешно проводиться без помощи разработчика ОО и с минимальными затратами. Анализ поддерживается независимым тестированием ФБО.

ОУД 2 предусматривает *структурное тестирование*. ОУД 2 содержит требование сотрудничества с разработчиком для получения информации о проекте и результатах тестирования без существенного увеличения стоимости или затрат времени. Анализ поддерживается независимым тестированием ФБО, свидетельством разработчика об испытаниях, основанных на функциональной спецификации, выборочным независимым подтверждением результатов тестирования разработчиком, анализом стойкости функций и свидетельством поиска разработчиком явных уязвимостей (например, из общедоступных источников).

ОУД 3 предусматривает *методическое тестирование и проверку*. ОУД 3 позволяет достичь максимального доверия путем применения надлежащего проектирования безопасности без значительного изменения существующей практики качественной разработки. Предполагается проведение всестороннего исследования ОО и процесса его разработки без существенных затрат на изменение технологии проектирования. Анализ поддерживается независимым тестированием ФБО, свидетельством разработчика об испытаниях, основанных на функциональной спецификации и проекте верхнего уровня, выборочным независимым подтверждением результатов тестирования разработчиком, анализом стойкости функций и свидетельством поиска разработчиком явных уязвимостей (например, из общедоступных источников).

ОУД 4 предусматривает *методическое проектирование, тестирование и просмотр*. ОУД 4 применим, когда разработчикам или пользователям требуется независимо получаемый уровень доверия от умеренного до высокого в ОО общего назначения и имеется готовность нести дополнительные, связанные с безопасностью производственные затраты. Это самый высокий уровень, на который обычно экономически целесообразно ориентироваться для существующих типов продуктов. Анализ поддерживается независимым тестированием ФБО,

свидетельством разработчика об испытаниях, основанных на функциональной спецификации и проекте верхнего уровня, выборочным независимым подтверждением результатов тестирования разработчиком, анализом стойкости функций, свидетельством поиска разработчиком уязвимостей и независимым анализом уязвимостей, демонстрирующим противодействие попыткам проникновения нарушителей с низким потенциалом нападения.

ОУД 5 предусматривает *полуформальное проектирование и тестирование*. ОУД 5 применим, когда разработчикам или пользователям требуется независимо получаемый высокий уровень доверия для запланированной разработки со строгим подходом к разработке, не влекущим излишних затрат на применение узко специализированных методов проектирования безопасности. Тем самым, предполагается, что ОО будут проектироваться и разрабатываться с намерением достичь ОУД 5. Анализ поддерживается независимым тестированием ФБО, свидетельством разработчика об испытаниях, основанных на функциональной спецификации, проекте верхнего уровня и проекте нижнего уровня, выборочным независимым подтверждением результатов тестирования разработчиком, анализом стойкости функций, свидетельством поиска разработчиком уязвимостей и независимым анализом уязвимостей, демонстрирующим противодействие попыткам проникновения нарушителей с умеренным потенциалом нападения. Анализ также включает проверку правильности анализа разработчиком скрытых каналов [92].

ОУД 6 предусматривает *полуформальную верификацию проекта и тестирование*. ОУД6 позволяет разработчикам достичь высокого доверия путем применения специальных методов проектирования безопасности в строго контролируемой среде разработки с целью получения высококачественного ОО для защиты высоко оцениваемых активов от значительных рисков. Анализ поддерживается независимым тестированием ФБО, свидетельством разработчика об испытаниях, основанных на функциональной спецификации, проекте верхнего уровня и проекте нижнего уровня, выборочным независимым подтверждением результатов тестирования разработчиком, анализом стойкости функций, свидетельством поиска разработчиком уязвимостей и независимым анализом уязвимостей, демонстрирующим противодействие попыткам проникновения нарушителей с высоким потенциалом нападения. Анализ также включает проверку правильности систематического анализа разработчиком скрытых каналов.

ОУД 7 предусматривает *формальную верификацию проекта и тестирование*. ОУД7 применим при разработке безопасных ОО для исполь-

зования в ситуациях чрезвычайно высокого риска и/или там, где высокая ценность активов оправдывает более высокие затраты. Практическое применение ОУД7 в настоящее время ограничено ОО, которые строго ориентированы на реализацию функциональных возможностей безопасности и для которых возможен подробный формальный анализ. Анализ поддерживается независимым тестированием ФБО, свидетельством разработчика об испытаниях, основанных на функциональной спецификации, проекте верхнего уровня, проекте нижнего уровня и представлении реализации, полным независимым подтверждением результатов тестирования разработчиком, анализом стойкости функций, свидетельством поиска разработчиком уязвимостей и независимым анализом уязвимостей, демонстрирующим противодействие попыткам проникновения нарушителей с высоким потенциалом нападения. Анализ также включает проверку правильности систематического анализа разработчиком скрытых каналов.

Сводное описание оценочных уровней доверия приведено в таблице 2.13. Все уровни являются иерархически упорядоченными, и каждый ОУД представляет более высокое доверие, чем любой из

Таблица 2.13

Сводное описание оценочных уровней доверия

Класс доверия	Семейство доверия	Компоненты доверия из оценочного уровня доверия						
		ОУД1	ОУД2	ОУД3	ОУД4	ОУД5	ОУД6	ОУД7
Управление конфигурацией	ACM_AUT				1	1	2	2
	ACM_CAP	1	2	3	4	4	5	5
	ACM_SCP			1	2	3	3	3
Поставка и эксплуатация	ADO_DEL		1	1	2	2	2	3
	ADO_IGS	1	1	1	1	1	1	1
Разработка	ADV_FSP	1	1	1	2	3	3	4
	ADV_HLD		1	2	2	3	4	5
	ADV_IMP				1	2	3	3
	ADV_INT					1	2	3
	ADV_LLD				1	1	2	2
	ADV_RCR	1	1	1	1	2	2	3
	ADV_SPM				1	3	3	3

Окончание табл. 2.13

Класс доверия	Семейство доверия	Компоненты доверия из оценочного уровня доверия						
		ОУД1	ОУД2	ОУД3	ОУД4	ОУД5	ОУД6	ОУД7
Руководства	AGD_ADM	1	1	1	1	1	1	1
	AGD_USR	1	1	1	1	1	1	1
Поддержка жизненного цикла	ALC_DVS			1	1	1	2	2
	ALC_FLR							
	ALC_LCD				1	2	2	3
	ALC_TAT				1	2	3	3
Тестирование	ATE_COV		1	2	2	2	3	3
	ATE_DPT			1	1	2	2	3
	ATE_FUN		1	1	1	1	2	2
	ATE_IND	1	2	2	2	2	2	3
Оценка уязвимостей	AVA_CCA					1	2	2
	AVA_MSU			1	2	2	3	3
	AVA_SOF		1	1	1	1	1	1
	AVA_VLA		1	1	2	3	4	4

предыдущих. Увеличение доверия от ОУД к ОУД достигается заменой какого-либо компонента доверия иерархически более высоким компонентом из того же семейства доверия (т.е. увеличением строгости, области и/или глубины оценки) и добавлением компонентов доверия из других семейств доверия (т.е. добавлением новых требований).

Помимо заявленных в части 3 ОК ОУД, можно представлять другие комбинации компонентов доверия. Операция *усиления* оценочных уровней доверия допускает добавление компонентов доверия (из семейств доверия, до этого не включенных в ОУД) или замену компонентов доверия в ОУД другими, иерархически более высокими компонентами доверия из того же самого семейства доверия. Исключение из ОУД какого-либо составляющего его компонента доверия является недопустимым. В случае, если производится усиление ОУД, необходимо строго обосновать полезность и дополнительную ценность добавленного к ОУД компонента доверия. ОУД может быть также расширен за счёт применения требований доверия, сформулированных в явном виде.

2.7.4. Общая методология оценки безопасности информационных технологий

Наибольший интерес среди сопутствующих ОК материалов представляет ГОСТ Р ИСО/МЭК 18045–2008 «Информационная технология. Методы и средства обеспечения безопасности. Методология оценки безопасности информационных технологий», более известный как «Общая методология оценки» (ОМО). Документ охватывает все виды деятельности по оценке, соответствующие классам доверия из части 3 ОК, входящим в оценочные уровни доверия 1–4, кроме связанных с оценкой профилей защиты и заданий по безопасности.

«Общая методология оценки» описывает минимум действий, выполняемых оценщиком при проведении оценки по ОК, с использованием критериев и свидетельств оценки, определенных в ОК. Документ предназначен, прежде всего, для оценщиков, использующих ОК, и экспертов органов по сертификации, подтверждающих действия оценщиков.

Взаимосвязь между ОМО и 3-ей частью ОК показана на рис. 2.8.

Тем самым, ОМО в основном представляет собой детализированную последовательность действий оценщика при проведении проверок по каждому из компонентов доверия части 3 ОК для ОУД 1–4. Для более высоких ОУД методология считается в настоящее время неопределённой.

Процесс оценки состоит из выполнения оценщиком задачи получения исходных данных для оценки, задачи оформления результатов оценки и подвидов деятельности по оценке.

По каждому элементу действий оценщик выносит *вердикт — положительный* или *отрицательный. Общий вердикт* является положитель-

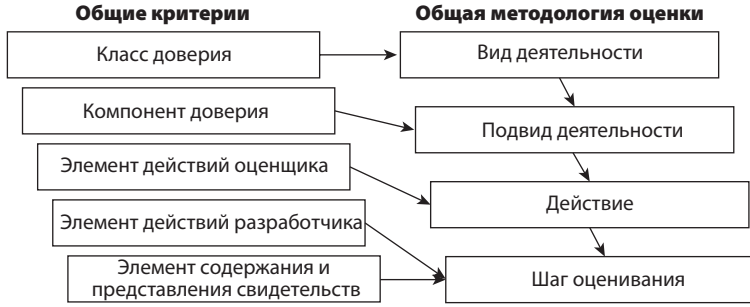

Рис. 2.8. Взаимосвязь между ОК и ОМО

ным тогда и только тогда, когда все составляющие вердикта положительные.

Результаты оценки оформляются в виде *технического отчёта об оценке* (ТОО), имеющего следующую структуру:

Введение, включающее:

— идентификаторы системы сертификации;

— идентификаторы контроля конфигурации ТОО (название, дата, номер версии и т.д.);

— идентификаторы контроля конфигурации ЗБ и ОО;

— ссылка на ПЗ;

— идентификатор разработчика;

— идентификатор заявителя;

— идентификатор оценщика.

— Описание архитектуры ОО, представляющее высокоуровневое описание ОО и его главных компонентов, основанное на проекте верхнего уровня.

1. Оценка, включающая:

— методы, технологии, инструментальные средства и стандарты, применяемые при оценке;

— сведения об ограничениях, принятых при проведении оценки;

— правовые аспекты оценки, заявления о конфиденциальности и т.д.

2. Результаты оценки, где для каждого вида деятельности приводятся:

— название рассматриваемого вида деятельности;

— вердикт, сопровождаемый обоснованием, для каждого компонента доверия, определяющего этот вид деятельности, как результат выполнения соответствующего действия ОМО и составляющих его шагов оценивания.

3. Выводы и рекомендации:

— общий вердикт;

— рекомендации, которые могут быть полезны для органа по сертификации.

4. Перечень свидетельств оценки, где для каждого использованного при проведении оценки свидетельства указываются:

— составитель;

— название;

— уникальная ссылка.

5. Перечень сокращений и глоссарий терминов.

6. Сообщения о проблемах:

— полный перечень сообщений о проблемах;

— их текущее состояние.

Сообщения о проблемах (СП) представляют собой механизм для запроса разъяснений или для определения проблемы по тому или иному аспекту оценки. При отрицательном вердикте оценщик обязан представить СП для отражения результата оценки. При этом СП должны иметь следующую структуру:

— идентификатор оцениваемого ОО;

— задача/подвид деятельности по оценке, при выполнении которой/которого проблема была выявлена;

— суть проблемы;

— оценка ее серьезности (например, приводит к отрицательному вердикту, задерживает выполнение оценки или требует решения до завершения оценки);

— организация, ответственная за решение вопроса;

— рекомендуемые сроки решения;

— влияние на оценку отрицательного результата решения проблемы.

Адресаты рассылки СП и процедуры обработки сообщения определяются правилами, действующими в системе сертификации.

В заключение темы по «Общим критерим», следует отметить, что основным недостатком ОК является объем и сложность разработки и восприятия документации по сравнению с традиционными подходами. В связи с эти можно рекомендовать изучение учебного примера ЗБ, представленного по адресу: www.cnpo.ru/zb.pdf.

2.8. Современные нормативные документы ФСТЭК России

Ранее мы отмечали, что на сегодняшний день при проведении сертификационных испытаний СЗИ по требованиям безопасности информации в абсолютном большинстве случаев используются традиционные руководящие документы Гостехкомиссии России (см. подр. 2.4). Несмотря на некоторое моральное устаревание данных документов, применение «Общих критериев» при проведении сертификационных испытаний до последнего времени было весьма ограниченно и носило скорее экспериментальный характер.

В то же время перспективы развития оценочных стандартов в данной области сейчас связаны, прежде всего, с «Общими критериями». В настоящее время в ФСТЭК России разработаны документы,

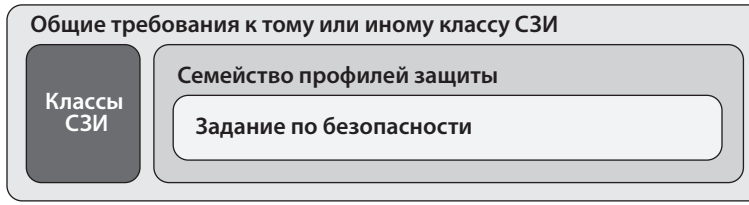

Рис. 2.9. Структура требований для средств защит информации

касающиеся требований к системам обнаружения вторжений и антивирусных средств. Ожидается, что в ближайшее время на базе «Общих критериев» будут разработаны требования к средствам доверенной загрузки, средствам двухфакторной аутентификации, средствам контроля съемных носителей информации, средствам предотвращения утечек информации и др.

В общем случае, стандарты нового поколения будут иметь структуру, представленную на рис. 2.9.

Для каждого из основных классов средств защиты информации будет разработан документ, классифицирующий данный вид средств защиты и определяющий требования к соответствующему семейству профилей защиты. В свою очередь, профили защиты должны служить основанием для разработки заданий по безопасности, на соответствие которым и будет сертифицироваться конечный продукт.

На момент написания данной монографии ФСТЭК России сформулировал требования к системам обнаружения вторжений (СОВ) и системам антивирусной защиты (САВЗ). Документы уже вступили в силу в 2012 г., но имеют пометку «для служебного пользования». Профили защиты СОВ и САВЗ, предназначенных для защиты информации, не содержащей сведений, составляющих государственную тайну, являются общедоступными и в настоящее время размещены на официальном сайте ФСТЭК России [69].

Ожидается, что новые требования к другим классам СЗИ будут иметь аналогичную структуру.

2.8.1. Требования к системам обнаружения вторжений

В нормативном документе ФСТЭК России «Требования к системам обнаружения вторжений» под *системами обнаружения вторжений* понимаются программные и программно-аппаратные технические средства, реализующие функции автоматизированного обнаружения в информационных системах действий, направленных на преднаме-

ренный несанкционированный доступ к информации, а также специальных воздействий на информацию в целях её добывания, уничтожения, искажения или блокирования. Система обнаружения вторжений рассматривается как один из базовых элементов системы защиты информационной системы.

В соответствии с общепринятой практикой [9], в документе выделяются *два типа* систем обнаружения вторжений: это системы обнаружения вторжений уровня сети и системы обнаружения вторжений уровня узла. Основной задачей *системы обнаружения вторжений уровня сети* является сбор информации о сетевом трафике, передаваемом в пределах информационной системы, и ее дальнейший анализ с целью выявления вторжений. *Система обнаружения вторжений уровня узла* должна обнаруживать вторжения на основе анализа данных с узлов контролируемой информационной системы, включающих: сетевой трафик, проходящий через контролируемые узлы, события, регистрируемые в журналах аудита операционной системы и прикладного программного обеспечения, вызовы функций, обращения к ресурсам.

Для каждого из типов выделяются 6 классов защиты систем обнаружения вторжений, требования ужесточаются от шестого класса к первому. Каждому классу защиты соответствует определенная категория информационных систем:

— СОВ, соответствующие 6 классу защиты, применяются в информационных системах персональных данных 3 и 4 классов;

— СОВ, соответствующие 5 классу защиты, применяются в информационных системах персональных данных 2 класса;

— СОВ, соответствующие 4 классу защиты, применяются в информационных системах персональных данных 1 класса, информационных системах общего пользования II класса, а также в государственных информационных системах, в которых обрабатывается информация ограниченного доступа, не содержащая сведений, составляющих государственную тайну;

— СОВ, соответствующие 3, 2 и 1 классам защиты, применяются в информационных системах, в которых обрабатывается информация, содержащая сведения, составляющих государственную тайну.

Состав функциональных требований к системам обнаружения вторжений является достаточно традиционным. Помимо непосредственно возможностей по выявлению, анализу и реагированию на те или иные события, предъявляются требования к системе управления параметрами СОВ (табл.2.14).

Таблица 2.14

Функциональные требования безопасности для СОВ уровня сети 4 класса

Компонент	Название компонента
FAU_GEN.1	Генерация данных аудита
FAU_GEN.2	Ассоциация идентификатора пользователя
FAU_SAR.1	Просмотр аудита
FAU_SAR.2	Ограниченный просмотр аудита
FAU_SAR.3	Выборочный просмотр аудита
FMT_MOF.1	Управление режимом выполнения функций безопасности
FMT_MTD.1	Управление данными функций безопасности СОВ
FMT_MTD.2	Управление ограничениями данных функций безопасности СОВ
FMT_SMR.1	Роли безопасности
FPT_TST.1	Тестирование функций безопасности СОВ
FID_COL_EXT.1	Сбор данных о сетевом трафике
FID_ANL_EXT.1	Базовый анализ данных СОВ
FID_MTH_EXT.1	Методы анализа
FID_MTH_EXT.2	Детализация эвристического метода анализа
FID_RCT_EXT.1	Базовое реагирование СОВ
FID_PCL_EXT.1	Анализ протоколов
FID_CON_EXT.1	Механизмы администрирования
FID_UPD_EXT.1	Обновление базы решающих правил СОВ
FID_INF_EXT.1	Интерфейс СОВ

Из табл. 2.14 видно, что часть ФТБ сформулирована в явном виде (имеют постфикс «EXT»). Остальные требования — разработаны на основе стандартных ФТБ, приведенных во второй части стандарта ГОСТ Р ИСО/МЭК 15408.

Новый нормативный документ устанавливает требования доверия, сформулированные на основе предопределенных в третьей части стандарта ГОСТ Р ИСО/МЭК 15408 оценочных уровней доверия (ОУД):

— СОВ 6 класса защиты должны соответствовать ОУД 1+ (усиленный);

— СОВ 5 класса защиты должны соответствовать ОУД 2+;

— СОВ 4 класса защиты должны соответствовать ОУД 3+;

— для СОВ 3, 2 и 1 классов защиты предъявляются требования более высоких ОУД.

Надо заметить, заявитель должен разработать и реализовать значительное количество технологических процедур, обеспечивающих обновление баз решающих правил СОВ. Например, перечень процедур для систем 4 класса представлен в табл. 2.15.

Интерес вызывает уточнение стандартных (приведенных в третьей части стандарта ГОСТ Р ИСО/МЭК 15408) требований доверия для обеспечения преемственности требованиям по контролю отсутствия недекларированных возможностей, изложенных в руководящем документе Гостехкомиссии России «Защита от несанкционированного доступа к информации. Часть 1. Программное обеспечение средств защиты информации: Классификация по уровню контроля отсутствия недекларированных возможностей» (см. подр.2.4.4). Например, для 4-го класса защиты разработчик должен обеспечить представление реализации для всех функций безопасности СОВ на уровне исходных текстов всего программного обеспечения, входящего в состав СОВ, а также указать в документации значения контрольных сумм файлов, входящих в состав СОВ. При этом, испытательная лаборатория должна сделать независимое заключение, что представление реализации — точное и полное отображение функциональных требований безопасности СОВ, в том числе на основе результатов контроля исходного состояния программного обеспечения и контроля полноты и отсутствия избыточности исходных текстов на уровне файлов.

Важно отметить, что в документе впервые в отечественной практике в явном виде допускается обновление баз решающих правил разработчиком. При этом разработчик ежегодно предоставляет в испытательную лабораторию, проводившую испытания соответствующей системы, подробный отчет обо всех внесенных изменениях и об их возможном влиянии на безопасность системы. Такой подход позволяет значительно ускорить процедуру обновления по сравнению с традиционным, требовавшим в некоторых случаях проведения инспекционного контроля после внесения каждого изменения в изделие.

В то же время жесткая формализация требований к свидетельствам оценки накладывает определенные обязательства на систему

Таблица 2.15

Технологические процедуры для СОВ 4 класса

Наименование процедуры	Краткое описание процедуры
Уникальная маркировка	Процедура уникальной маркировки каждого сертифицированного изделия
Управление конфигурацией	Система управления конфигурацией, отслеживающая: представление реализации изделия, проектную документацию, тестовую документацию, документацию пользователя, документацию администратора и документацию управления конфигурацией.
Поставка системы обнаружения вторжений пользователю в соответствии с разработанной процедурой	Процедуры, необходимые для поддержки безопасности при распространении версий к местам использования.
Обновления базы решающих правил	Фиксация момента получения нового типа вторжения. Выпуск обновления базы сигнатур за заданное время. Уведомление об обновлении базы сигнатур. Поставка обновления базы сигнатур. Контроль целостности обновлений базы данных решающих правил. Представление обновлений для проведения внешнего контроля. Анализ влияния обновлений на безопасность системы обнаружения вторжений.

менеджмента качества заявителя: перечень документов, которые ему придется разрабатывать и предоставлять для оценки выглядит достаточно внушительным. Так, например, для 4 класса защищенности необходимо разработать следующие документы:

— задание по безопасности;

— руководство по установке;

— функциональная спецификация;

— анализ соответствия между всеми смежными парами имеющихся представлений функций безопасности;

— руководство администратора;

— руководство пользователя;

— результаты анализа стойкости функции безопасности;

— описание процедуры фиксации момента появления нового типа вторжения;

— описание процедуры выпуска обновления базы сигнатур за заданное время;

— описание процедуры уведомления об обновлении базы сигнатур;

— описание процедуры поставки обновления базы сигнатур;

— описание процедуры контроля целостности обновлений базы решающих правил;

— описание процедуры представления обновлений для проведения внешнего контроля;

— методика анализа влияния обновлений на безопасность системы обнаружения вторжений.

2.8.2. Требования к средствам антивирусной защиты

В нормативном документе ФСТЭК России «Требования к средствам антивирусной защиты» под *средствами антивирусной защиты* (САВЗ) понимаются программные средства, используемые в целях обеспечения защиты информации, и реализующие функции обнаружения компьютерных программ либо иной компьютерной информации, предназначенных для несанкционированного уничтожения, блокирования, модификации, копирования компьютерной информации или нейтрализации средств защиты информации (вредоносные компьютерные программы, компьютерные вирусы), а также реагирования на обнаружение этих программ и информации. Нормативный документ выделяет четыре типа САВЗ:

— САВЗ типа «А», предназначенные для централизованного администрирования САВЗ, установленными на компонентах информационных систем (серверах, АРМ);

— САВЗ, типа «Б», предназначенные для применения на серверах информационных систем;

— САВЗ, типа «В», предназначенные для применения на АРМ информационных систем;

— САВЗ, типа «Б», предназначенные для применения на автономных АРМ.

Для каждого из типов выделяются 6 классов защиты САВЗ, требования ужесточаются от шестого класса к первому. Каждому классу защиты соответствует определенная категория информационных систем:

— САВЗ, соответствующие 6 классу защиты, применяются в информационных системах персональных данных 3 и 4 классов;

— САВЗ, соответствующие 5 классу защиты, применяются в информационных системах персональных данных 2 класса;

— САВЗ, соответствующие 4 классу защиты, применяются в информационных системах персональных данных 1 класса, информационных системах общего пользования II класса, а также в государственных информационных системах, в которых обрабатывается информация ограниченного доступа, не содержащая сведений, составляющих государственную тайну;

— САВЗ, соответствующие 3, 2 и 1 классам защиты, применяются в информационных системах, в которых обрабатывается информация, содержащая сведения, составляющих государственную тайну.

Рассматриваемый нормативный документ определяет требования ко всем 24-м классам САВЗ, в нем в явном виде сформулированы все функциональные требования и требования доверия, которые должны войти в соответствующие ПЗ и, в дальнейшем, в ЗБ на конкретные изделия.

По аналогии с требованиями к СОВ, в состав функциональных требований безопасности к САВЗ, помимо непосредственно возможностей по выявлению и удалению компьютерных вирусов, входят требования по обновлению базы данных признаков компьютерных вирусов и требования к системе управления параметрами САВЗ. Важной особенностью документа является то, что помимо собственно требований по обнаружению компьютерных вирусов, предъявляются требования к методам такого обнаружения (сигнатурный, эвристический).

Нормативный документ устанавливает требования доверия, сформулированные на основе предопределенных в 3-ей части стандарта ГОСТ Р ИСО/МЭК 15408 оценочных уровней доверия (ОУД):

— САВЗ 6 класса защиты должны соответствовать ОУД 1+ (усиленный),

— САВЗ 5 класса защиты — ОУД 2+,

— САВЗ 4 класса защиты — ОУД 3+,

— для САВЗ 3, 2 и 1 классов защиты предъявляются требования более высоких ОУД.

Как и в случае с СОВ, заявитель должен разработать и реализовать значительное количество технологических процедур и документов, обеспечивающих безопасную (доверенную) разработку САВЗ. Надо заметить, что при проведении сертификации для 4 класса защиты и выше разработчик должен представить в испытательную лабораторию представление реализации функций безопасности САВЗ — исходные тексты ПО.

В документе в явном виде допускается обновление базы данных признаков компьютерных вирусов (данное положение представлено

в виде требований доверия, сформулированных в явном виде). При этом разработчик ежегодно предоставляет в испытательную лабораторию, проводившую испытания САВЗ, подробный отчет обо всех внесенных изменениях и об их возможном влиянии на безопасность системы. Такой подход позволяет ускорить процедуру обновления по сравнению с традиционным, требовавшим в некоторых случаях проведения инспекционного контроля после внесения каждого изменения в изделие.

3. МЕТРИКИ И МОДЕЛИ ИСПЫТАНИЙ

3.1. Показатели и метрики испытаний

3.1.1. Виды показателей объекта испытаний

При выполнении оценки соответствия по требованиям безопасности информации используются полуколичественные и количественные показатели. Полуколичественными показателями обычно выступают частные показатели, оцениваемые по некоторой бальной шкале. Например, при сертификационных испытаниях на соответствие традиционным РД используются частные показатели положительного результата проверок, принимающие значения, скажем, {0, 1}.

Количественные показатели могут принимать различные точные числовые значения. Примером использования таких показателей является проведение тематических исследований и сертификационных испытаний на соответствие ТУ, сертификационных испытаний по надежности обработки информации, обеспечению полноты, безошибочности, актуальности и защищенности информации в процессе функционирования информационных систем и другие [16, 64, 93].

Значения показателей могут быть, например, определены экспертным, регистрационным или расчетным путем.

В подразделах 2.4–2.8 рассмотрены полуколичественные показатели защищенности информации. В таблицах 3.1 и 3.2 приведены примеры показателей качества, регламентированные национальными стандартами для программных и автоматизируемых систем: ГОСТ 28195 и ГОСТ 15987 [26].

Допускаются различные подходы к формированию комплексных показателей, обычно, путем получения полиномиальной зависимости от частных показателей (метрик):

Таблица 3.1

Частные показатели «технологической безопасности» по ГОСТ 28195

Наименование	Метод оценки
Наличие требований к программе по устойчивости функционирования при наличии ошибок во входных данных	Экспертный
Возможность обработки ошибочных ситуаций	Экспертный
Полнота обработки ошибочных ситуаций	Экспертный
Наличие тестов для проверки допустимых значений входных данных	Экспертный
Наличие системы контроля полноты входных данных	Экспертный
Наличие средств контроля корректности входных данных	Экспертный
Наличие средств контроля непротиворечивости входных данных	Экспертный
Наличие требований к программе по восстановлению процесса выполнения в случае сбоя операционной системы, процессора, внешних устройств	Экспертный
Наличие требований к программе по восстановлению результатов при отказах системы	Экспертный
Наличие средств восстановления процесса в случае сбоев оборудования	Экспертный
Наличие возможности разделения по времени выполнения отдельных функций программ	Экспертный
Наличие возможности повторного старта с точки останова	Экспертный
Наличие проверки параметров и адресов по диапазону их значений	Экспертный
Наличие обработки граничных результатов	Экспертный
Наличие обработки неопределенностей (деление на 0, квадратный корень из отрицательного числа и т.д.)	Экспертный

Продолжение табл. 3.1

Наименование	Метод оценки
Наличие централизованного управления процессами, конкурирующими из-за ресурсов	Экспертный
Наличие возможности автоматически обходить ошибочные ситуации в процессе вычисления	Экспертный
Наличие средств, обеспечивающих завершение процесса решения в случае помех	Экспертный
Наличие средств, обеспечивающих выполнение программы в сокращенном объеме в случае ошибок или помех	Экспертный
Показатель устойчивости к искажающим воздействиям	$P(Y) = 1 - D/K$, где: D — число экспериментов, в которых искажающие воздействия приводили к отказу, K- число экспериментов, в которых имитировались искажающие воздействия
Вероятность безотказной работы	$P = 1 - Q/N$, где: Q — число зарегистрированных отказов, N — число экспериментов
Оценка по среднему времени восстановления	$$Q_{\text{Е}} = \begin{cases} 1, \text{ если } T_{\text{в}} \le T_{\text{в}}^{\text{доп}} \\ \dfrac{T_{\text{в}}^{\text{доп}}}{T_{\text{в}}}, \text{ если } T_{\text{в}} > T_{\text{в}}^{\text{доп}}, \end{cases}$$ где: $T_{\text{в}}^{\text{доп}}$ — допустимое среднее время восстановления; $T_{\text{в}}$ — среднее время восстановления, которое определяется по формуле: $$T_{\text{в}} = \frac{1}{N}\sum_{i}^{N} T_{\text{в}_i},$$ где: N — число восстановлений; $T_{\text{в}_i}$ — время восстановления после i-го отказа

Окончание табл. 3.1

Наименование	Метод оценки
Оценка по продолжительности преобразования входного набора данных в выходной	$Q_{\text{п}_i} = \begin{cases} 1, \text{ если } T_{\text{п}_i} \le T_{\text{п}_i}^{\text{доп}} \\ \dfrac{T_{\text{п}_i}^{\text{доп}}}{T_{\text{п}_i}}, \text{ если } T_{\text{п}_i} > T_{\text{п}_i}^{\text{доп}}, \end{cases}$ где: $T_{\text{п}_i}^{\text{доп}}$ – допустимое время преобразования *i*-го входного набора данных; $T_{\text{п}i}$ – фактическая продолжительность преобразования *i*-го входного набора данных

Таблица 3.2

Типовая номенклатура показателей автоматизированных систем [26]

Характеристики качества функционирования АС	Основные показатели качества функционирования АС, для которых должны быть заданы допустимые значения
Надежность представления запрашиваемой или выдаваемой принудительно информации (выполнения задаваемых технологических операций)	Средняя наработка объекта на отказ или сбой – $T_{\text{нар}}$
	Среднее время восстановления объекта после отказа или сбоя – $T_{\text{вос}}$
	Коэффициент готовности объекта – $K_{\text{г}}$
	Вероятность надежного представления и/или доведения запрашиваемой (выдаваемой принудительно) выходной информации $P_{\text{над}}$ в течение заданного периода функционирования АС $T_{\text{зад}}$
	Вероятность надежного выполнения технологических операций $P_{\text{над}}$ в течение заданного периода функционирования АС $T_{\text{зад}}$

Окончание табл. 3.2

Характеристики качества функционирования АС	**Основные показатели качества функционирования АС, для которых должны быть заданы допустимые значения**
Своевременность представления запрашиваемой или выдаваемой принудительно информации (выполнения задаваемых технологических операций)	Среднее время реакции системы при обработке запроса и/или доведении информации $T_{\text{полн}}$ или вероятность своевременной обработки информации $P_{св}$ за заданное время $T_{зад}$ Среднее время выполнения технологической операции $T_{полн}$ или вероятность выполнения технологической операции $P_{св}$ за заданное время $T_{зад}$
Полнота используемой информации	Вероятность обеспечения полноты оперативного отражения в АС новых реально существующих объектов предметной области — $P_{полн}$
Актуальность используемой информации	Вероятность сохранения актуальности информации на момент ее использования $P_{акт}$
Безошибочность информации после контроля	Вероятность $P_{бум}$ после отсутствия ошибок во входной информации на бумажном носителе при допустимом времени на процедуру контроля $T_{зад}$ Вероятность $P_{маш}$ после отсутствия ошибок во входной информации на машинном носителе при допустимом времени на процедуру контроля $T_{зад}$
Корректность обработки информации	Вероятность $P_{корр}$ получения корректных результатов обработки информации за заданное время $T_{зад}$
Конфиденциальность информации	Вероятность сохранения конфиденциальности информации $P_{конф}$ в течение периода ее объективной конфиденциальности $T_{конф}$
Безошибочность действий должностных лиц	Вероятность безошибочных действий должностных лиц $P_{чел}$ в течение заданного периода функционирования АС $T_{зад}$
Защищенность от опасных программно-технических воздействий	Вероятность отсутствия опасного воздействия $P_{возд}$ в течение заданного периода функционирования АС $T_{зад}$
Защищенность от НСД	Вероятность сохранения защищенности от НСД информационных и программных ресурсов АС $P_{НСД}$

$$P = \sum_{i=1}^{k} b_i p_i + \sum_{i \neq j}^{k} b_{ij} p_i p_j + \sum_{i=1}^{k} b_{ii} p_i^2 + \ldots,$$

где: b_i — коэффициент i-й метрики, k — число метрик.

ГОСТ 28195 регламентирует линейную модель с весовыми коэффициентами иерархических показателей: «фактор-критерий-метрика» (рис. 3.1).

Следует сказать, что в области тестирования ПО измеряемые количественные частные показатели принято называть метриками.

Обычно выделяют три типа метрик:

— метрики сложности программного кода;
— метрики покрытия программного кода;
— метрики полноты функционального тестирования.

3.1.2. Метрики сложности программного кода

Метрики сложности программного кода являются измеримыми количественными характеристиками особенностей реализации программ [25]. Фактически, эти метрики позволяют получить идентификационный профиль конкретных программ при статическом анализе. На практике это позволяет решить задачи аутентификации ПО, оценить сложность ПО и, как следствие, уровень безошибочности программного проекта, трудоемкость анализа и доработок ПО, стоимость и сроки работ, эффективность технологии разработки и внедрения и др. Часто метрики являются параметрами моделей планирования испытаний, которые рассмотрены в разделе 3.2.4 [49].

В настоящее время метрики сложности программного кода условно можно разделить на следующие:

— меры длины («объема») программного обеспечения;
— метрики сложности текста программ;
— метрики сложности по управлению;
— метрики сложности по данным;
— объектно-ориентированные метрики;
— интегральные метрики, включающие вышеназванные.

Меры длины кода являются самыми популярными метриками при предварительной оценке стоимости работ. Основными из них являются:

— размер дистрибутива в байтах,

[25] IEEE Std. 1061–1998.

Рис. 3.1. Формирование показателя соответствия реальных и декларированных возможностей по ГОСТ 28195

— размер исходного кода в байтах;

— число строк исходного кода, включая комментарии (SLOC, source lines of code);

— число операторов (SI, source instructions);

— число функциональных объектов;

— число операторов в эквивалентных командах ассемблера (AELOC, assembly-equivalent lines of code).

Очень важно, что метрики исходного кода указываются относительно конкретного языка программирования.

При проведении анализа программного кода меры длины мало информативны — для этого используются другие классы метрик.

К **метрикам сложности текста** (линейной сложности) программ относят измерительные характеристики ПО, касающиеся учета сочетаний команд программы без анализа ее графа. Наиболее знаменитыми являются измерительные метрики Холстеда (Halsted) [82]:

– длина программы $N = N_1 + N_2$, где N_1 – число операторов и N_2 – операндов;

– объем программы (в бит информации) $V = N \log_2(\eta)$, где $\eta = \eta_1 + \eta_2$ – словарь операторов и операндов языка программирования.

Используя аппарат теории информации и гипотезу Страуда об умственных операциях, Холстед предложил ряд оценочных метрик ПО, как-то:

– теоретическая длина программы $\tilde{N} = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2$;

– теоретический объем программы $\tilde{V} = \tilde{N} \log_2(\eta)$;

– уровень программы $\tilde{L} = \frac{2}{\eta_1} \frac{\eta_2}{N_2}$;

– трудоемкость кодирования $D = 1 / \tilde{L} = \frac{\eta_1}{2} \frac{N_2}{\eta_2}$;

– интеллектуальное содержание программы $I = \tilde{L}V = \frac{2}{\eta_1} \frac{\eta_2}{N_2} N \log_2(\eta)$;

– сложность программы $E = V / L = \frac{\eta_1 N_2 N \log_2 \eta}{2\eta_2}$;

– время реализации программы (в секундах) $\tilde{T} = \frac{E}{18} = \frac{\eta_1 N_2 N \log_2 \eta}{36\eta_2}$;

– число программных ошибок $\hat{B} = \frac{V}{3000} = \frac{N \log_2(\eta_1 + \eta_2)}{3000}$.

Помимо метрик Холстеда, к метрикам линейной сложности относят всевозможные количественные показатели числа разных операторов, уровней вложенности, а также комментарий. Например, метрики Джилба (Gilb) определяют насыщенность программы и максимальную вложенность операторами цикла и условными операторами.

К **метрикам сложности по управлению** (основанным на управляющем графе программ) прежде всего относят цикломатическое число МакКейба (McCabe), характеризующее число независимых маршрутов в программе:

$$V(G) = e - n + 2,$$

где: e — количество дуг; n — количество вершин управляющего графа программы.

Цикломатическая сложность программного комплекса, содержащего вызовы подпрограмм, определяется как сумма цикломатических сложностей самой программы и вызываемых подпрограмм [26].

Существует множество модификаций метрики МакКейба, касающихся добавления взвешенных коэффициентов сложности линейных участков, учета уровня вложенности и т.д.

К другим популярным метрикам, основанным на графах разного рода, относят:

— количество пересечений дуг управляющего графа программы, известное как метрика Вудворда (Woodward),

— число возможных путей, известное как метрика Шнейдевинда (Schneidewind),

— цикломатическое число сети Петри,

— число вершин графа параллельности,

— число висячих вершин,

— разница числа входов и выходов вершин управляющего графа и другие.

К **метрикам сложности по данным** (основанным на потоке данных) можно отнести:

— «спен» — число утверждений, содержащих переменную между ее первым и последним появлением в тексте программы;

— число используемых глобальных переменных в модуле;

— число неиспользуемых информационных объектов;

— соотношение разного рода типов связей по данным между модулями (подпрограммами);

— соотношение функциональных назначений переменных и др.

Наиболее известным примером последней группы метрик является эмпирическая метрика Чепена (Chepen), вычисляемая следующим образом:

$$Q = P + 2M + 3C + 0{,}5T,$$

где: P — вводимые переменные для расчетов и для обеспечения вывода; M — модифицируемые или создаваемые внутри программы переменные; C — управляющие переменные; T — неиспользуемые переменные.

[26] NIST SP 500–235: 1996.

Объектно-ориентированные метрики отражают принципиальные особенности парадигмы объектно-ориентированного программирования и проектирования. Наиболее популярными являются метрики Ли (Li), а также Чидамбера (Chidamber) и Кемерера (Kemerer), например:

— количество локальных методов (NLM, number of local methods);

— количество новых методов (NNM, number of new methods);

— количество общедоступных методов в классе (NPM, number of public methods);

— количество абстрактных классов (NAC, number of abstract classes);

— количество потомков классов (NDC, number of descendent classes);

— количество реакций на класс (RFC, responce for class) — число методов, которые могут выполнить какое-либо действие в ответ на сообщение, отправленное объектом в этом классе, используя один уровень вложенности;

— отсутствие единства методов (LCOM, lack cohesion of methods) — число непересекающихся пар методов (не имеющих общих переменных) за минусом количества подобных пар метода;

— связь через абстрактный тип данных (CTA, coupling through abstract data type) — число классов, которые используются в качестве абстрактных типов данных в декларации класса;

— связь через сообщения (CTM, coupling through message passing) — число различных сообщений, отправленных классом к другим классам, за исключением сообщений, отправленных объектам, созданным как локальные объекты в локальных методах класса.

В заключение подраздела следует сказать, что не существует универсальных метрических характеристик ПО — в зависимости от решающих задач рассматриваются метрики, являющиеся значимыми для программного проекта [18]. Известно достаточное большое число анализаторов кода [23,71,74,], которые позволяют рассчитать метрики ПО, например, анализатор безопасности программного кода «АК-ВС» формирует:

— количество функциональных объектов в проекте;

— количество связей функциональных объектов по управлению;

— количество связей функциональных объектов по информации;

— общее количество связей функциональных объектов;

— количество висящих функциональных объектов в проекте;

— количество информационных объектов в проекте;
— количество ветвей кода в проекте;
— цикломатическая сложность кода по МакКейбу;
— объем программы по Холстеду;
— количество файлов в проекте;
— количество найденных потенциально опасных конструкций;
— количество файлов с потенциально опасными конструкциями;
— среднее количество объектов в файле;
— средний размер файла;
— объем исходных текстов (Кбайт);
— количество строк кода в проекте.

3.1.3. Метрики покрытия программного кода

Метрики покрытия программного кода используются при структурном тестировании (по методу «белого ящика») с целью подтверждения требуемой полноты проверок и контроля эффективности процесса тестирования, а также выявления наиболее слабо проверенных модулей и участков кода, тупиковых и избыточных фрагментов и других ошибок структуры и логики работы ПО.

Количественная мера покрытия кода часто является частным показателем качества ПО, например:

$$Q = P \times Cov,$$

где: P — показатель степень безошибочности кода; Cov — показатель покрытия кода [53].

К наиболее известным метрикам покрытия кода относят следующие:

— покрытие конструкций;
— покрытие ветвлений условий;
— покрытие условий;
— покрытие веток и условий;
— покрытие маршрутов;
— покрытие потока данных.

Покрытие конструкций (операторов) кода — метрика позволяет понять, была ли вызвана каждая из исполняемых конструкций кода (команда).

Частным случаем метрики покрытия конструкций является метрика покрытия **базовых блоков,** когда в качестве единиц кода выступает каждая последовательность конструкций без ветвления. За-

метим, что понятие «базовый блок» не совпадает с понятием «ветвь», которая всегда начинается с условного оператора[27].

Одним из основных преимуществ метрики покрытия конструкций кода является то, что она может быть применена непосредственно к объектному (бинарному) коду, и при этом ей же можно пользоваться без обработки исходных текстов. По этой причине она получила широкое распространение и в других областях (например, в профилировке кода).

Главный недостаток метрики покрытия конструкций — это отсутствие «чувствительности» к некоторым управляющим структурам и логическим операциям. Например, покрытие операторов не позволяет определить, достигли ли циклы условий своего завершения — оно лишь показывает, был ли факт хотя бы однократного выполнения тела цикла.

Метрика **покрытия ветвлений** (decision[28]) определяет, были ли проверены (как на значение *true*, так и на *false*) булевы выражения в управляющих структурах (таких как выражения *if* или *else*). Всё булево выражение целиком считается одним предикатом, который может быть либо истинным, либо ложным, вне зависимости от того, содержит ли он операторы логического-И или логического-ИЛИ. Можно встретить альтернативные названия метрики: покрытие ветвей (branch), покрытие ребер, покрытие дуг.

Основное преимущество этой метрики — простота и отсутствие проблем с покрытием операторов.

Ее же недостатком является то, что метрика игнорирует ветви внутри булевых выражений, которые происходят в случае использования сокращенной схемы вычислений.

Метрика **покрытия по условиям** возвращает вывод «истина» или «ложь» для каждого условия в программе. Условие — это операнд логического оператора, который не содержит внутри себя других логических операторов. Покрытие по условиям измеряет условия независимо друг от друга.

Эта метрика во многом схожа с покрытием ветвей условий, но она имеет более высокую чувствительность к потоку выполнения.

В стандартах[29] разделяют покрытие ветвлений и покрытие условий, считая первое более слабым. Однако полное покрытие по усло-

---

[27] IEEE Std 100–1992.

[28] RTCA/DO-178BB. FAA, 1992.

[29] Там же.

виям не гарантирует полного покрытия ветвей условий (из проблемы «мертвого кода»).

Метрика **покрытия по веткам и условиям** — это гибридный метод, полученный на основе объединения покрытия ветвлений и покрытия условий. Его основное преимущество — это простота и отсутствие ограничений его компонентных метрик.

Метрика **покрытия маршрутов** выдает информацию о том, были ли пройдены все возможные маршруты в каждой функции. Маршрутом называют уникальную последовательность ветвей логических условий (branches) от входа в функцию до выхода из неё. Эта метрика также известна под названием «покрытие предикатов».

У покрытия маршрутов имеются два принципиальных недостатка. Первым является то, что число маршрутов экспоненциально зависит от числа ветвей. Например, функция, содержащая 10 выражений типа ***if,*** будет требовать для тестирования 1024 маршрута. Добавление хотя бы ещё одного выражения *if* удваивает их число до 2048. Второй недостаток — многие маршруты просто невозможно выполнить из-за связей по информации.

Еще один недостаток касается циклов. Поскольку циклы приводят к неограниченному количеству маршрутов, данная метрика рассматривает лишь ограниченное количество возможных итераций. Однако имеется огромное количество вариаций этой метрики, работающих с циклами. Например, **граничное тестирование маршрутов**, которое предполагает лишь две ситуации для циклов в программе: нулевое число повторов и отличное от нуля число повторов. Для циклов *do-while* возможны следующие два варианта: с одной итерацией и большим, чем одна итерация.

Метрику **покрытия потока данных** относят к вариации покрытия маршрутов, когда рассматриваются только подмаршруты от присваивания переменных до последующих ссылок на эти переменные.

Преимуществом этой метрики является то, что обнаруженные с помощью нее маршруты имеют непосредственное отношение к режиму, в котором программа обрабатывает данные. Основной её недостаток — сложность реализации.

Кроме перечисленных фундаментальных метрик, известны метрики, используемые на практике при решении отдельных задач:

- метрика покрытия функций (процедур);
- метрика покрытия вызовов;
- метрика покрытия циклов;
- метрика покрытия операторов сравнения;

— метрика покрытия по гонкам;
— метрика покрытия таблиц (массивов).

Мы можем сравнить относительные преимущества показателей, при этом более сильная метрика включает в себя результаты более слабой метрики.

Например, покрытие ветвлений условий включает в себя покрытие конструкций, поскольку выполнение каждой ветви должно, по идее, привести к выполнению каждой её конструкции. Но это утверждение справедливо, только когда поток выполнения непрерывен до самого конца всех базовых блоков. Например, C/C++ функция может никогда не вернуть управление вызвавшему её базовому блоку, поскольку внутри неё используются ***throw, abort***, семейство вызовов ***exec, exit*** или ***long jmp.***

Покрытие по ветвям/условиям включает в себя покрытие ветвлений условий и покрытие по условиям (по определению). Покрытие маршрутов включает в себя покрытие ветвей условий.

Анализ покрытия кода является одним из методов структурного тестирования, которые помогают обеспечить целостность и отсутствие пропусков в тестовом наборе. При анализе кода продуктов, разработанных на самых распространенных языках программирования (C/C++ и Java) из всех метрик наиболее универсальной и эффективной в целом показало себя покрытие по ветвям/условиям.

Разумеется, из-за чрезвычайно высокой сложности ПО контроль покрытия кода ограничивается разумной долей детализации.

3.1.4. Метрики полноты функционального тестирования

Метрики полноты функционального тестирования (по методу «черного ящика») раскрывают содержание тестовых процедур, с одной стороны, и позволяют детально оценить уровень завершенности тестирования в процессе работ — с другой.

При проведении функционального тестирования ПО полнота тестирования, как правило, оценивается по следующим параметрам:
— полнота покрытия требований;
— подсистем (модулей) ПО;
— полнота функциональных возможностей;
— полнота пространства входных параметров.

Полнота покрытия требований описывается следующей формулой:

$$M_{\text{тр}} = \frac{R_{\text{т}}}{R},$$

где: R_τ — число требований, предъявляемых к ПО, проверенных в ходе тестирования набором тестовых процедур, R — общее число требований, предъявляемых к ПО.

В ходе разработки набора тестовых процедур $\{\tau_1, \tau_2, ..., \tau_i, ..., \tau_n\}$ выполняется сопоставление требования $r_j \in R$, предъявляемого к ПО, и тестовых процедур, которые тестируют выполнение данного требования. Для наглядности данное сопоставление представляют в форме матрицы трассировки: в заголовках колонок матрицы (табл. 3.3) расположены требования r_j, а в заголовках строк — процедуры тестирования τ_i. На пересечении расположена отметка, означающая, что требование текущей колонки покрыто процедурой тестирования текущей строки.

В качестве **метрики полноты покрытия подсистем (модулей)** ПО можно выбрать следующее соотношение**:**

$$M_{\text{м}} = \frac{S_\tau}{S},$$

где: S_τ — число подсистем (модулей) ПО, проверенных в ходе тестирования набором тестовых процедур, S — общее число подсистем (модулей, функциональных объектов) ПО. Предполагается, что тестируемое ПО состоит из совокупности подсистем (например, подсистема контроля целостности), а подсистемы — из модулей (например, *a.exe*, *b.dll*). Для наглядности данное сопоставление представляют в форме матрицы покрытия тестовыми процедурами (табл. 3.4, m_i^j — *i*-й модуль *j*-й подсистемы).

Полнота покрытия функциональных возможностей описывается следующей формулой:

$$M_{\text{ф}} = \frac{F_\tau}{F},$$

где: F_τ — число функциональных возможностей ПО, проверенных в ходе тестирования набором тестовых процедур $\{\tau_1, \tau_2, ..., \tau_i, ..., \tau_n\}$; F — общее число функциональных возможностей ПО. По аналогии с метрикой покрытия требований можно ввести понятие матрицы трассировки функциональных возможностей (табл. 3.5): в заголовках колонок таблицы расположены функциональные возможности f_j, а в заголовках строк — процедуры тестирования τ_j.

Рассмотрим **метрику полноты покрытия входных параметров.** Пусть $A_P = \{\alpha_1, \alpha_2, ..., \alpha_s\}$ — множество параметров, при-

Таблица 3.3

Матрица трассировки требований

Тестовая процедура	Требование			
	r_1	...	r_j	...
τ_1	×		×	
...				
τ_j			×	
...				

нимаемых на вход тестируемым ПО P (факторы тестирования), а $B_P = \{\beta_1, \beta_2, ..., \beta_g\}$ — множество выходных параметров тестируемого ПО. Каждый параметр может принимать одно из нескольких значений: $\alpha_i \in D^P_{\alpha_i}$, $\beta_j \in D^P_{\beta_j}$, где $D^P_{\alpha_i}$ и $D^P_{\beta_j}$ — области допустимых значений входного α_i и выходного β_j параметров соответственно. При тестировании на вход ПО подается кортеж $\tilde{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_s) \in D^P_{\alpha_1} \times D^P_{\alpha_2} \cdots \times D^P_{\alpha_s}$ и выполняется сравнение фактического поведения ПО $\tilde{\beta} = (\beta_1, \beta_2, ..., \beta_g) = P(\tilde{\alpha}) \in D^P{}_{\beta_1} \times D^P{}_{\beta_2} ... \times D^P{}_{\beta_g}$ с ожидаемым $\tilde{\beta}_o$, определенным в спецификации.

Полноту покрытия пространства входных параметров можно оценить следующим образом:

$$M_{\text{вх}} = \frac{\left|\tilde{D}^P_{\alpha_1}\right|\left|\tilde{D}^P_{\alpha_2}\right| ... \left|\tilde{D}^P_{\alpha_s}\right|}{\left|D^P_{\alpha_1}\right|\left|\tilde{D}^P_{\alpha_2}\right| ... \left|\tilde{D}^P_{\alpha_s}\right|},$$

где: $\left|\hat{D}^P_{\alpha_i}\right|$ — мощность множества значений входного параметра αi, покрытого при выполнении тестирования, $\left|D^P{}_{\alpha_i}\right|$ — мощность множества значений входного параметра α_i.

Таким образом, при выполнении полного тестирования на вход ПО должно быть подано $\left|D^P{}_{\alpha_1}\right| \cdot \left|D^P{}_{\alpha 2}\right| \cdot ... \cdot \left|D^P{}_{\alpha_s}\right|$ кортежей $\tilde{\alpha}$. Для оценки сложности проблемы тестирования положим $\left|D^P_{\alpha_1}\right| = \left|D^P_{\alpha_2}\right| = ... = \left|D^P_{\alpha_s}\right| = N$, тогда мощность пространства входных значений равна N^s. Пусть некоторая программа принимает на вход $s = 34$ бинарных ($N = 2$) значения. При проведении тестирования на всем пространстве значений входных параметров на вход ПО необхо-

Таблица 3.4

Матрица покрытия подсистем (модулей)

Тестовая процедура	Объект тестирования								
	Подсистема 1			Подсистема 2		...	Подсистема *h*		
	m_1^1	...	m_k^1	m_2^1	...	...	m_1^m	...	m_d^h
τ_1	×					×	×		
τ_i			×					×	
...		×			×				×

Таблица 3.5

Матрица трассировки функциональных возможностей

Тестовая процедура	Требование			
	f_1	...	f_j	...
τ_1	×		×	
...		×		
τ_j	×		×	
...				

дим подать $2^{34} = 1{,}7 \cdot 10^{10}$ кортежа. Для современного ПО проведение тестирования на всем пространстве значений входных параметров (исчерпывающее тестирование) является практически неразрешимой задачей.

К основным путям решения данной проблемы следует отнести следующие.

1. Тестирования с использованием случайных значений входных параметров. Значение компонент α_i входного кортежа $\tilde{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_s)$ генерируются случайным образом из областей допустимых значений $D_{\alpha_i}^P$. Тестирование заканчивается при достижении приемлемого покрытия пространства входных значений.

2. Тестирования с использованием разбиения области допустимых значений на эквивалентные подмножества. Области допустимых значений $D_{\alpha_i}^P$ компонент входного кортежа разбиваются на подмножества, значения которых считаются эквивалентными

с точки зрения функциональных особенностей тестируемого ПО: $D^P_{\alpha_i} = D^P_{\alpha_{i1}} \cup D^P_{\alpha_{i2}} \cup \ldots \cup D^P_{\alpha_{iE}}$. При тестировании на вход ПО в качестве компонента кортежа подается один представитель подмножества $D^P_{\alpha_{il}}$.

3. Тестирование граничных значений. При использовании данной стратегии в первую очередь тестируются граничные значения областей допустимых значений $D^P_{\alpha_i}$ компонент входного кортежа.

4. Тестирование методом комбинаторного покрытия. При использовании метода *t*-факторного комбинаторного покрытия выполняется генерация входных кортежей, покрывающих все возможные значения подкортежей из *t* компонент[30].

3.2. Модели оценки технологической безопасности и планирования испытаний

Одним из путей повышения уровня безопасности ПО является использование на этапах тестирования и испытаний ПО математических моделей, позволяющих получить достоверные оценки показателей безопасности ПО и эффективности технологии его разработки. Большинство таких моделей заимствованы из теории надежности технических систем, поэтому в литературе их часто называют моделями надежности ПО [72, 85]. С точки зрения испытаний программных средств по требованиям безопасности информации, понятие «надежность функционирования ПО» эквивалентно понятию «технологической безопасности ПО», где под ошибкой понимается *уязвимость* (дефект, недекларированная возможность), потенциально влияющая на безопасность системы и инфраструктуры [17, 44].

Опыт испытательных лабораторий показывает, что применение математических моделей не должно отвлекать экспертов от реального трудоемкого и ответственного процесса исследования ПО и должно, главным образом, способствовать принятию правильных решений. Поэтому модели целесообразно классифицировать по целевому признаку и имеющимся входным статистикам, получаемым на различных этапах жизненного цикла ПО, а именно на следующие:

1. Отладочные модели, позволяющие оценить показатели технологической безопасности ПО в зависимости от прогонов на заданных областях входных данных и последующих доработок ПО;

[30] NIST SP 800–53A: 2010; NIST SP 800–142: 2010.

2. Временные модели роста надежности, позволяющие оценить показатели технологической безопасности ПО в зависимости от времени испытаний;

3. Модели полноты тестирования, позволяющие получить оценки показателей доверия к процессу оценки соответствия ПО;

4. Модели сложности ПО, позволяющие оценить метрики сложности ПО и связанные с ними показатели качества и безопасности ПО.

3.2.1. Отладочные модели программ

В основе отладочных моделей лежит утверждение, что свойство надежности программы меняется только при ее доработках, а измеряется путем прогона программы на заданных областях входных данных. Такие модели часто называют моделями надежности, основанными на областях входных данных (input-domain models).

Условно отладочные модели можно разделить на модели, ориентированные на величину доработки [53], и модели, ориентированные на покрытие входных данных [70,78]. Приведем примеры подобных моделей.

3.2.1.1. Немонотонная модель отладки и обновлений программного обеспечения

В основе модели положена гипотеза, что степень надежности ПО при его изменениях может как повышаться, так и понижаться (рис. 3.2):

$$P_u = P_0 + \sum_{j=1}^{u} \Delta P_j,$$

где: u — число проведенных доработок ПО, ΔP_j — приращение степени надежности после j-й доработки.

Изменение степени надежности ПО после j-й доработки можно представить линейным оператором:

$$\Delta P_j = A_j \left(1 - P_{j-1}\right) - B_j P_{j-1},$$

где: P_{j-1} — вероятность безошибочной работы ПО после $(j-1)$-й доработки; $(1 - P_{j-1})$ — вероятность обнаружения ошибок ПО после $(j-1)$-й доработки; A_j — коэффициент эффективности доработки, характеризующий уменьшение вероятности ошибки за счет j-й доработки; B_j —

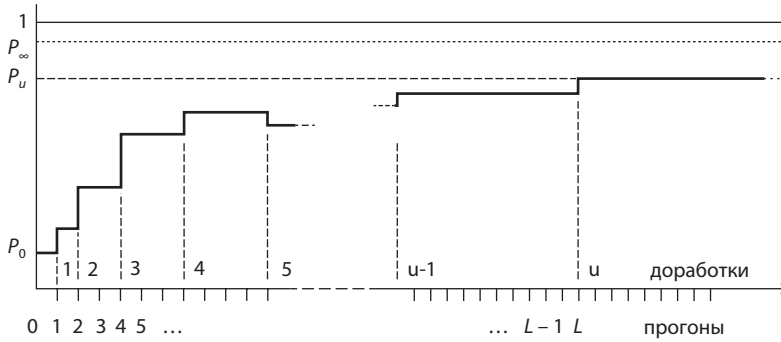

Рис. 3.2. Изменение степени надежности по результатам доработок

коэффициент негативности доработки, характеризующий снижение степени надежности за счет *j*-й доработки.

Перейдя к рекуррентному выражению и считая предельную степень надежности равной $P_\infty = \dfrac{A_j}{A_j + B_j}$, можно получить модель оценки надежности ПО:

$$P_u = P_\infty - \left(P_\infty - P_0\right)\prod_{j=1}^{u}\left(1 - A_j / P_\infty\right).$$

Для учета различной степени эффективности доработки можно ввести метрику величины измененного кода *kij*, например, при исправлении ошибок и при обновлении ПО. Это позволяет получить основное расчетное выражение для показателя надежности ПО:

$$P_u = P_\infty - \left(P_\infty - P_0\right)\prod_{j=1}^{u}\left(1 - \sum_{i=1}^{2} a_i k_{ij} / P_\infty\right),$$

где: α_i — коэффициент эффективности доработки ПО с целью исправления ошибки или обновления; P_0 — начальная степень надежности; P_∞ — предельная степень надежности.

Данная модель зависит от 4-х параметров (P_0, P_∞, α_1, α_2), расчет которых удобно осуществить, например, с помощью метода максимального правдоподобия [53].

Приведем пример расчета параметров модели. В качестве исходной статистики можно использовать множества испытаний $\{n_j\}$ и

отказов $\left\{\hat{m}_j\right\}$ между доработками, а также множество метрик по доработкам $\{k_{ij}\}$. Тогда при допущениях о независимости прогонов ПО функция максимального правдоподобия представляет собой вероятность получения общей выборки ($n_i, \hat{m}_j, j = \overline{1,u}$) числа отказов в проведенных сериях прогонов ПО:

$$L_u = \prod_{j=1}^{u} C_{m_j}^{n_j} P_j^{n_i - \hat{m}_j} (1 - P_j)^{\hat{m}_j}.$$

Для удобства можно прологарифмировать и преобразовать функцию L_u к следующему виду:

$$L_u = \sum_{J=1}^{u} \left(\hat{m}_j \ \ln \left(1 - P_\infty + \left(P_\infty - P_0 \right) \prod_{l=1}^{j} \left(1 - \sum_{i=1}^{2} \frac{a_i k_{ij}}{P_\infty} \right) \right) + \right.$$

$$\left. + \ (n_j - \hat{m}_j) \ \ln \left(P_\infty - (P_\infty - P_0) \prod_{l=1}^{j} (1 - \sum_{i=1}^{2} \frac{a_i k_{ij}}{P_\infty}) \right) \right).$$

Полученная функция является выпуклой и задана на выпуклом множестве. Поэтому, для нахождения максимума функции правдоподобия можно, например, использовать модифицированный метод наискорейшего спуска с переменным параметром шага h:

$$\left.\begin{aligned} P_0^{r+1} &= P_0^r + h \left(\frac{\partial \ \ln L\left(P_0^r, P_\infty^r, a_1^r, a_2^r\right)}{\partial P_0} \right) \\ P_\infty^{r+1} &= P_\infty^r + h \left(\frac{\partial \ \ln L\left(P_0^{r+1}, P_\infty^r, a_1^r, a_2^r\right)}{\partial P_\infty} \right) \\ a_1^{r+1} &= a_1^r + h \left(\frac{\partial \ \ln L\left(P_0^{r+1}, P_\infty^{r+1}, a_1^r, a_2^r\right)}{\partial a_1} \right) \\ a_2^{r+1} &= a_2^r + h \left(\frac{\partial \ \ln L\left(P_0^{r+1}, P_\infty^{r+1}, a_1^{r+1}, a_2^r\right)}{\partial a_2} \right) \end{aligned}\right\},$$

где: r — номер итерации.

Следует сказать, что модель позволяет получить формулы для планирования испытаний, например, рассчитать число j необходимых доработок ПО для достижения требуемой степени надежности $P_{\text{тр}}$:

$$j = ceil\left(\frac{\ln\left(\frac{P_\infty - P_{тр}}{P_\infty - P_u}\right)}{\ln\left(1 - a / P_\infty\right)}\right),$$

где: a — усредненный коэффициент эффективности доработки ПО.

Считая, что при доработках не вносятся дополнительные ошибки, т.е. $B_j = 0$, можно получить формулу числа оставшихся ошибок после u-й доработки:

$$N_u = ceil\left(\frac{\ln(\frac{1 - P_0}{1 - P_u})}{\ln(1 - a)}\right).$$

3.2.1.2. Структурная модель Нельсона

Модель, получившая название модели Нельсона (Nelson) [78], является биноминальной моделью Бернулли с наложенными правилами по использованию входных данных.

В частности, область входных данных ПО задается в виде k непересекающихся областей — $\{Z_i\}$, которым однозначно соответствует множество вероятностей $\{p_i\}$ того, что соответствующий набор данных будет выбран для очередного прогона ПО. Таким образом, если при выполнении N_i прогонов программы (на Z_i наборе входных данных) n_i из них закончились отказом, то степень надежности функционирования ПО определяется выражением:

$$P = 1 - \sum_{i=1}^{k} \frac{n_i}{N_i} p_i.$$

Модель позволяет рассчитать вероятность P_u безотказного выполнения n прогонов программы:

$$P_u = \prod_{j=1}^{u}(1 - Q_j) = e^{\left\{\sum_{j=1}^{u} \ln\left(1 - Q_j\right)\right\}},$$

где: $Q_j = \sum_{i=1}^{k} p_{ji}\chi_i$; χ_i — характеристическая функция отказа на i-ом наборе данных; p_{ji} — вероятность появления i-го набора в j-ом прогоне.

Заметим, в структурной модификации модели Нельсона предлагается для нахождения p_{ji} использовать анализ графа ПО. К сожалению, проведение анализа структурно-сложного модифицируемого ПО для решения указанных задач на практике не представляется эффективным[31].

Можно продемонстрировать переход от моделей отладки к временным моделям. Полагая, что Δt_j — время выполнения j-го прогона, можно получить следующее расчетное выражение:

$$P_u = e^{(\sum_{j=1}^{u} \lambda(t_j)\Delta t_j)},$$

где: $\lambda\left(t_j\right) = \dfrac{-\ln(1-Q_j)}{\Delta t_j}$ — интенсивность отказов; $t_j = \sum_{i=1}^{j} \Delta t_i$ — суммарное время выполнения j прогонов ПО.

Полагая, что Δt_i становится относительно малой величиной с ростом u-числа испытаний, имеем известную формулу безотказной работы технических средств (экспоненциальная временная NHPP-модель роста надежности):

$$P(t) = e^{-\int_0^t \lambda(z)dz},$$

где: $\lambda(z)$ — функция риска.

Несмотря на очевидную адекватность моделей отладки процессу доработки ПО, к общим недостаткам моделей отладки относят требования по большому количеству испытаний для получения точных оценок. Однако данная категория трудностей решается путем применения статистического метода Вальда [37,76].

3.2.2. Модели роста надежности от времени

Модели роста надежности (reliability growth model) относят к вероятностным динамическим моделям дискретных систем с непрерывным или дискретным временем [12,21,43,49,76,85,90]. Большинство популярных моделей данного класса можно свести к Марковским однородным, неоднородным или полумарковским моделям массового обслуживания.

В однородных Марковских моделях полагается, что общее число ошибок ПО — неизвестная конечная постоянная величина. Число

[31] IEEE Guide to SWEBOK, 2004.

ошибок, оставшихся в ПО в процессе тестирования и отладки, описывается экспоненциальным законом распределения. Интенсивность ошибок $\lambda(i)$ зависит от текущего i-состояния системы и не зависит от прошлых состояний.

По сравнению с однородными Марковскими моделями, в полумарковских моделях полагается, что $\lambda(t_i)$ интенсивность ошибок зависит не только от числа оставшихся ошибок в ПО, но и от t_i времени в этом состоянии.

В настоящее время все более популярным классом временных моделей становятся неоднородные Марковские модели. В данных моделях общее число ошибок ПО является случайной величиной, описываемой Пуассоновским законом распределения, причем интенсивность потока ошибок не является линейной функцией от времени. По этой причине модели часто называют Пуассоновскими (Non-Homogeneous Poisson Process model, NHPP-модели). В зависимости от вида функции интенсивности Пуассоновского потока NHPP-модели разделяют на выпуклые, S-образные, бесконечные.

Существуют модификации моделей надежности ПО путем применения Байесовского подхода, которые иногда выделяют в отдельный класс моделей [99].

Следует сказать, что для расчета параметров динамических моделей традиционно используется метод максимального правдоподобия, в редких случаях — методы линейной регрессии.

Приведем примеры самых популярных временных моделей роста надежности каждого вида.

3.2.2.1. Экспоненциальная модель роста надежности программ

Экспоненциальная модель роста надежности, получившая название модели Елинского-Моранды (Jelinski-Moranda, JM-модель), основана на допущениях, что в процессе тестирования ПО длительность интервалов времени между обнаружением двух ошибок имеет экспоненциальное распределение с интенсивностью отказов, пропорциональной числу необнаруженных ошибок. Все ошибки одного типа, равновероятны и независимы друг от друга. Каждая обнаруженная ошибка мгновенно устраняется, число оставшихся ошибок уменьшается на единицу[32].

Согласно указанным предположениям интенсивность ошибок в JM-модели имеет следующий вид:

[32] IEEE Std. 1633–2008.

$$\lambda_i = \phi\left(N - \left(i-1\right)\right),$$

где: N — число ошибок, первоначально присутствующих в программе; $(i-1)$ — число обнаруженных ошибок.

Функция плотности распределения времени обнаружения i-й ошибки, отсчитываемого от момента выявления $(i-1)$-й ошибки, имеет вид:

$$f\left(t_i\right) = \lambda_i e^{-\lambda_i t_i} = \phi(N-(i-1))e^{-\phi(N-(i-1))t_i},$$

где: ϕ — коэффициент пропорциональности, интерпретируемый как интенсивность выявления ошибок; N — число ошибок, первоначально присутствующих в программе; t_i — интервал между $(i-1)$-й и i-й ошибками.

Полученное выражение позволяет, например, найти формулы для вероятности безошибочной работы, вероятности устранения всех ошибок за заданное время [76], средней наработки на ошибку, среднего времени устранения всех ошибок:

$$P\left(t_i\right) = \int_0^{t_i} f\left(t\right)dt = e^{-\lambda_i t_i} = e^{-\phi(N-(i-1))t_i};$$

$$P_{\geq N}\left(t_{\text{зад}}\right) = 1 - N\sum_{i=1}^{N}\left(-1\right)^{N-i}\frac{C_{N-1}^{i-1}}{N-(i-1)}e^{-\lambda_i t_{\text{зад}}};$$

$$\bar{t} = \int_0^{\infty} tf\left(t\right)dt = \frac{1}{\lambda_i} = \frac{1}{\phi\left(N-\left(i-1\right)\right)};$$

$$\bar{t} = \frac{1}{\lambda_i} = \frac{1}{\phi}\sum_{i=1}^{N}\left(\frac{1}{i}\right).$$

JM-модель имеет два неизвестных параметра: N и ϕ. Приведем пример получения параметров модели, используя метод максимального правдоподобия. В качестве исходной статистики можно использовать множество интервалов времени между отказами $\left\{\hat{t}_i\right\}$. Тогда при допущениях о независимости данной выборки функция максимального правдоподобия представляет собой произведение функций плотностей $f(t_j)$:

$$L_u = \prod_{i=1}^{u} f\left(t_j\right) = \left(\phi\left(N-\left(i-1\right)\right)e^{-\phi\left(N-\left(i-1\right)\right)t_i}\right)\times$$
$$\times\prod_{i=1}^{u}\left(\phi\left(N-\left(i-1\right)\right)e^{-\phi\left(N-\left(i-1\right)\right)t_i}\right).$$

Для случая $u = N$ можно получить формулу:

$$L_N = N!\phi^N e^{-\phi\sum_{i=1}^{N}\left(N-(i-1)\right)t_i}.$$

Прологарифмировав функцию и найдя ее производные, получаем условия нахождения экстремума:

$$\begin{cases}\sum_{i=1}^{u}\left(\frac{1}{N-\left(i-1\right)}-\phi t_i\right)=0\\ \sum_{i=1}^{u}\left(\frac{1}{\phi}-\left(N-\left(i-1\right)\right)t_i\right)=0\end{cases}.$$

Решение системы уравнения можно найти численными методами.

В табл. 3.6 представлены примеры популярных Марковских моделей, причем, параметр N интерпретируется в качестве числа первоначальных ошибок в ПО [49].

Таблица 3.6

Марковские модели роста надежности программ

Название модели	**Интенсивность ошибок, λ_i**
JM-модель*	$\phi(N-(i-1))$
Модель Липова	$\phi(N-\sum_{j=1}^{i-1}N_j)$
Xui-модель	$\phi(e^{-k(N-i+1)}-1)$
Shanthikumar-модель	$\phi(N-(i-1))^k$
Bucchianico-модель	$1-\phi^{(N-(i-1))}$
JM-отладочная модель	$\phi(N-p(i-1))$
* IEEE Std. 1633–2008.	

3.3.2.2. Рэлеевская модель роста надежности программ

Развитием JM-модели является модель Шэка-Волвертона (Schick-Wolverton, SW-модель), в которой полагается, что интенсивность ошибок пропорциональна не только количеству необнаруженных ошибок в ПО, но и интервалу времени отладки:

$$\lambda_i = \phi\left(N-(i-1)\right)t_i,$$

где: N – число ошибок, первоначально присутствующих в программе; i – число обнаруженных ошибок; j – коэффициент пропорциональности, интерпретируемый как интенсивность выявления ошибок, t_i – интервал времени между (i–1) -й и i-й ошибками.

Отсюда выводится распределение Рэлея со следующей функцией плотности:

$$f\left(t_i\right) = \phi\left(N-(i-1)\right)t_i e^{-\phi\left(N-(i-1)\right)\frac{t_i^2}{2}},$$

причем, параметр релеевского распределения

$$\sigma_0 = 1/\sqrt{\phi\left(N-(i-1)\right)}.$$

Полученное выражение позволяет, например, найти формулы для вероятности безошибочной работы и средней наработки на ошибку:

$$P\left(t_i\right) = e^{-\left\{\frac{t_i^2}{2\sigma_0^2}\right\}} = e^{-\phi\left(N-(i-1)\right)\frac{t_i^2}{2}};$$

$$\overline{t} = \sqrt{\frac{\pi}{2}}\sigma_0 = \sqrt{\frac{\pi}{2\phi\left(N-(i-1)\right)}}.$$

Для расчета значений параметров N и ϕ модели по аналогии с JM-моделью удобно использовать метод максимального правдоподобия.

Примеры популярных полумарковских моделей представлены в таблице 3.7, причем параметр N также интерпретируется как число первоначальных ошибок в ПО [49].

3.3.2.3. Экспоненциальная NHPP-модель роста надежности программ

Первая NHPP-модель предложена Гоул и Окьюмотиу (Goel-Okumoto, GO-модель). В GO-модели полагается, что количество оши-

бок, проявляющихся в единицу времени,— это независимые случайные величины, распределенные по закону Пуассона с интенсивностью потока (функцией количества ошибок) $m(i)$, пропорциональной ожидаемому числу остающихся в программе ошибок на заданный момент времени.

Считая, что общее число ошибок конечно и равно a, имеем следующее дифференциальное уравнение:

$$m'\left(t\right) = ag - gm(t).$$

Решая указанное уравнение, можно получить основное расчетное выражение модели:

$$m(t) = a(1 - e^{(-gt)}),$$

где: a — коэффициент, характеризующий число ожидаемых ошибок в ПО (которые будут найдены в бесконечности); g— коэффициент интенсивности выявления ошибок.

Функция $m(t)$ показывает количество ошибок к моменту t, она выпуклая, монотонно возрастающая, стремящаяся к значению a- ожидаемому числу ошибок, которые будут выявлены при тестировании, т.е.: $\lim\limits_{t\to\infty}\left(m\left(t\right)\right) = a$.

Таблица 3.7

Полумарковские модели роста надежности программ

Название модели	**Интенсивность ошибок, λ_i**
SW-модель	$\phi(N-(i-1))t_i$
Гиперболическая модель	$\phi\left(N-\left(i-1\right)\right)(-at_i^2+bt_i+c)$
Sukert-модель	$\phi(N-(i-N_i))t_i$
Модифицированная модель Липова	$\phi\left(N-\sum\limits_{j=1}^{i-1}N_j\right)\left(\frac{t_i}{2}+\sum\limits_{j=1}^{i-1}t_i\right)$
SW-отладочная модель	$\phi(N-p(i-1))t_i$

Ожидаемое число ошибок, оставшихся к моменту *t*, можно рассчитать следующим образом:

$$\overline{m}\left(t\right) = m\left(\infty\right) - m\left(t\right) = ae^{-gt}.$$

Функция интенсивности возникновения ошибки (показывающая среднее число ошибок к моменту t) имеет следующий вид:

$$\lambda\left(t\right) = m'\left(t\right) = age^{-gt}.$$

Полученные выражения позволяют найти формулы для вероятностей того, что за заданное время будет выявлено и локализовано (или нет) то или иное количество ошибок, например:

$$P(t,k) = \frac{(m\left(t\right))^k}{k!}e^{-m(t)} = \frac{\left(a(1-e^{-gt})\right)^k}{k!}e^{-\left(a(1-e^{-gt})\right)},$$

где: k — число выявленных ошибок за заданное время *t*.

Параметры NHPP-модели можно найти с помощью метода максимального правдоподобия. Следует заметить, что если имеется статистика моментов t_i обнаружения ошибок, можно получить следующую функцию максимального правдоподобия:

$$L_u\left(t_1, t_2, \ldots t_u\right) = \prod_{j=1}^{u}\lambda(t_j)e^{-m(t_u)} = \prod_{j=1}^{u}age^{-gt_j}\,e^{-\left(a(1-e^{-gt_u})\right)}.$$

Для случая фиксации количества обнаруженных ошибок N_i на интервалах времени (t_{j-1}; t_j], мы имеем следующий вид функции максимального правдоподобия:

$$L_u\left(N_1, N_2, \ldots N_k\right) = \prod_{j=1}^{k}\frac{\left(m\left(t_j\right) - m\left(t_{j-1}\right)\right)^{N_j}}{N_j!}e^{-m(t_k)} =$$

$$= \prod_{j=1}^{k}\frac{\left(a(e^{-gt_{j-1}} - e^{-gt_j})\right)^{N_j}}{N_j!}e^{-\left(a(1-e^{-gt_k})\right)}.$$

3.3.2.4. S-образная NHPP-модель роста надежности программ

В настоящее время одной из самых популярных NHPP-моделей роста надежности является S-образная NHPP-модель Ямады (Yamada), в которой, в отличие от JM- и SW-подобных выпуклых моделей, делается дополнительное предположение о *S*-образной зависимо-

сти числа ошибок от времени тестирования. Понятийно *S*-образная зависимость числа обнаруженных ошибок от времени объясняется тем, что в начальной стадии тестирования имеется фаза изучения экспертом ПО.

Функция количества ошибок задается следующей формулой:

$$m(t) = a(1 - (1 + gt)e^{-gt}),$$

где: a — коэффициент, характеризующий число ожидаемых ошибок в ПО; g — коэффициент интенсивности выявления ошибок.

Ожидаемое число ошибок, оставшихся к моменту t, можно рассчитать следующим образом:

$$\overline{m}(t) = a(1 + gt)e^{-gt}.$$

Соответственно, интенсивность возникновения ошибки определяется следующим образом:

$$\lambda(t) = ag^2te^{-gt}.$$

Полученные выражения позволяют легко найти формулу для вероятностей того, что за заданное время будет выявлено и локализовано (или нет) то или иное количество ошибок. Параметры модели a и g можно найти с помощью метода максимального правдоподобия.

3.3.2.5. Степенная NHPP-модель роста надежности программ

В модели надежности, получившей название модели Дьюэйна (Duane), полагается, что функция количества ошибок задается формулой:

$$m(t) = \left(\frac{t}{a}\right)^g,$$

где: a — коэффициент, характеризуемый масштаб; g — коэффициент интенсивности выявления ошибок (степень).

Соответственно, интенсивность возникновения ошибки определяется следующим образом:

$$\lambda(t) = \frac{g}{a}\left(\frac{t}{a}\right)^{g-1}.$$

В модели если $g > 1$, то $\lim_{t\to\infty}(m(t)) = +\infty$, поэтому модель получила название бесконечной NHPP-моделью (NHPP-infinite).

Полученные выражения позволяют легко найти формулу для вероятностей того, что за заданное время будет выявлено и локализовано (или нет) то или иное количество ошибок. Параметры модели a и g можно найти с помощью метода максимального правдоподобия.

Примеры популярных NHPP-моделей представлены в таблице 3.8 [49].

Таблица 3.8

Неоднородные Марковские модели роста надежности программ

Название NHPP-модели	**Функция количества ошибок,** $m(t)$
Duane-модель*	at^g
Модель Гомпертца	ag^{c^t}
Goel-Okumoto – модель	$a(1-e^{-gt})$
Schneidewind-модель*	$a\,/\,g(1-e^{-gt})$
Модель Вейбулла	$a(1-e^{-gt^c})$
Yamada-экспоненциальная модель	$a(1-e^{(-rc(1-e^{(-gt)}))})$
S-образная модель Релея	$a(1-e^{(-rc(1-e^{(-\frac{gt^2}{2})}))})$
S-образная модель с задержкой*	$a(1-(1+gt)e^{-gt})$
S-образная модель с точкой перегиба	$\frac{a(1-e^{-gt})}{1+ce^{-gt}}$
Параметризованная S-изогнутая модель	$a\frac{1-e^{-gt}}{1+\psi(r)e^{-gt}}$
Dohiya – модель	$a(1-e^{-gt})\big/(1+e^{-gt})$
Модель Парето	$a(1-(1+t\,/\,c)^{1-g})$
Гиперэкспоненциальная модель	$a\left(1-\sum_{i=1}^{2}b_ie^{-g_it}\right)$

Окончание табл. 3.8

Littlewood –модель	$ac^g\left(\frac{1}{c^g}-\frac{1}{(c+t)^g}\right)$
Параболическая модель	$a\left(1-e^{-\left(\left(\frac{l}{3}\right)t^3+\left(\frac{m}{2}\right)t^2+nt\right)}\right)$
Логистическая модель	$\frac{a}{1+ke^{-gt}}$
Pham – модель	$a-ae^{-gt}(1+(g+d)t+gdt^2$
Zhang – модель	$\frac{a}{p-\beta}\left((1-\frac{(1+\alpha)e^{-gt}}{1+\alpha e^{-gt}})^{\frac{c}{g}(p-\beta)}\right)$
Xie-логарифмическая модель	$aln^g(1+t)$
Musa-Okumoto-логарифмическая модель*	$1/a\ln(agt+1)$
* IEEE Std. 1633–2008.	

3.3.2.6. Байесовская модель роста надежности программ

В рассмотренных ранее моделях надежности используются предположения о характере распределений априорно, причем сами параметры моделей имеют детерминированный вид. Для корректировки параметров предполагаемых распределений в зависимости от новых статистических данных (об обнаружении и исправлении ошибок ПО) можно использовать байесовский подход, заключающийся в интерпретации параметра модели надежности как случайной величины с априорной функцией распределения. В общем виде, согласно теореме Байеса, отношение апостериорной оценки параметров модели имеет вид:

$$h\left(X/t\right)=\frac{l(t/X)f(X)}{\int_{\Omega_x} l(t/X)f(X)dX},$$

где: $f(X)$ – априорная плотность распределения; X – вектор параметров априорного распределения; t – измеримые данные; $l(t/X)$ – функ-

ция правдоподобия; $h(X/t)$ — апостериорная оценка плотности распределения вектора X.

Надо понимать, что введение байесовского подхода существенно усложняет вычислительную сложность моделей. Наиболее простой байесовской модификацией является гамма-модель Литлвуда и Вэрала (Littlewood -Verrall, LV-модель).

Как и в JM-модели в LV-модели интервалы между ошибками подчинены экспоненциальному распределению с условной плотностью вероятности для интервала t_i между ошибками:

$$f\left(t_i \,/\, \lambda_i\right) = \lambda_i e^{-\lambda_i t_i}.$$

Рассматривая функцию интенсивности ошибок как стохастически убывающую, предлагается описать ее условную плотность вероятности гамма-распределением с параметрами $\psi(i)$ и α:

$$f\left(\lambda_i\right) = \frac{\left(\psi(i)\right)^{\alpha} \lambda_i^{\alpha-1} e^{-\psi(i)\lambda_i}}{\Gamma(\alpha)},$$

где: $\psi(i)$ — возрастающая функция, характеризующая уровень надежности; i — количество найденных ошибок; α — параметр, задающий форму кривой роста надежности; $\Gamma(\alpha)$ — гамма-функция Эйлера.

Кроме того, полагается, что безусловная плотность вероятности интервала t_i определяется распределением Парето, а априорная плотность вероятности параметра a является равномерной.

В LV-модели и ее модификациях предложено множество вариантов аппроксимации $\psi(i)$, наиболее популярной из которых является линейная: $\psi\left(i\right) \approx \beta_0 + \beta_1 i$, где i — количество найденных ошибок.

В последнем случае для вычисления 3-х неизвестных параметров $(\alpha, \beta_0, \beta_1)$ необходимо решить лишь следующую систему уравнений:

$$\begin{cases} n + \alpha \sum_{i=1}^{n} \left(\ln\left(\beta_0 + \beta_1 i\right) - \ln\left(t_i + \beta_0 + \beta_1 i\right) \right) = 0 \\ \alpha \sum_{i=1}^{n} \dfrac{1}{\beta_0 + \beta_1 i} - \left(\alpha + 1\right) \sum_{i=1}^{n} \dfrac{1}{t_i + \beta_0 + \beta_1 i} = 0 \\ \alpha \sum_{i=1}^{n} \dfrac{1}{\beta_0 + \beta_1 i} - \left(\alpha + 1\right) \sum_{i=1}^{n} \dfrac{i}{t_i + \beta_0 + \beta_1 i} = 0 \end{cases},$$

где: t_i — интервалы времени между ошибками; n — число обнаруженных ошибок.

Для случая линейной аппроксимации $\tilde{\psi}(i)$ искомую функцию интенсивности предлагается рассчитать следующим образом:

$$\tilde{\lambda}(t) = \frac{\alpha - 1}{\sqrt{\beta_0^2 + \beta_1 t(\alpha - 1)}}.$$

Модель позволяет также получить расчетные выражения для средней наработки на отказ и вероятности безотказной работы[33]:

$$\overline{t} = \frac{\tilde{\psi}(i)}{\tilde{\alpha} - 1};$$

$$P_i(t) = \left(\frac{\tilde{\psi}(i)}{(t + \tilde{\psi}(i))} \right)^{\tilde{\alpha}}.$$

3.2.3. Модели полноты тестирования

Модели оценки полноты тестирования ПО основаны на методах независимого внесения и выявления тестовых ошибок и методах проведения независимых экспертиз.

3.2.3.1. Модель учета внесенных ошибок

Модель учета внесенных ошибок, известная также как решение задачи теории вероятности «меченых рыб» или как модель Миллса (Mills), предполагает внесение в текст программы S тестовых ошибок. В процессе тестирования собирается статистика о выявленных ошибках: s внесенных и n реальных.

Предполагается, что тестовые ошибки вносятся случайным образом, выявление всех ошибок (внесенных и собственных) равновероятно. В этом случае, используя метод максимального правдоподобия, можно получить оценку числа первоначальных ошибок ПО:

$$N = \frac{Sn}{s},$$

где: S — число внесенных тестовых ошибок; s — число найденных внесенных ошибок; n — число найденных реальных ошибок.

Для обеспечения уверенности в результате модели (т.е. для расчета достаточного числа внесенных тестовых ошибок) предложена

[33] IEEE Std. 1633–2008.

следующая мера доверия при условии выявления в процессе тестирования всех S внесенных ошибок:

$$P_{NS} = \begin{cases} 1, \text{ при } n > N, \\ \dfrac{S}{S+n+1}, \text{ при } n \le N. \end{cases}$$

В табл.3.9 показаны примеры меры доверия для случая выявления всех тестовых ошибок.

Из формулы для вычисления меры доверия можно получить формулу для вычисления количества тестовых ошибок, которые необходимо внести в программу:

$$S = \frac{P_{NS}(N+1)}{1 - P_{NS}}.$$

Пример табличных значений для расчета количества тестовых ошибок, которые необходимо внести в программу для получения нужной уверенности в оценке, представлен в табл.3.10.

В случае, если в процессе испытаний выявлены не все внесенные тестовые ошибки, следует использовать следующую формулу:

$$P_{Ns} = \begin{cases} 1, \text{ при } n > N; \\ \dfrac{C_S^{s-1} n}{C_{S+N+1}^{N+n}}, \text{ при } n \le N, \end{cases}$$

где: s — число обнаруженных тестовых ошибок.

Данная модель носит интуитивный характер. На практике модель применяется для контроля эффективности работы экспертов,

Таблица 3.9

Мера доверия к модели Миллса

S	N										
	0	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
0	0	0	0	0	0	0	0	0	0	0	0
10	91	48	32	24	20	16	14	12	11	10	9
20	95	65	49	39	33	28	25	22	20	18	17
30	97	73	59	49	42	37	33	30	27	25	23
40	98	78	66	56	49	44	40	36	33	31	28
50	98	82	70	62	55	50	45	41	38	35	33

Таблица 3.10

Количество тестовых ошибок

S	*N*										
	0	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
0	0	0	0	0	0	0	0	0	0	0	0
10	0	2	2	3	5	6	7	8	9	10	11
20	0	3	5	8	10	13	15	18	20	23	25
30	0	5	9	13	18	22	24	30	35	39	43
40	1	7	14	21	27	34	41	47	54	61	67
50	1	11	21	31	41	51	61	71	81	91	101

проводящих аудит безопасности кода. К основному недостатку модели относят проблему случайного внесения уязвимостей.

3.2.3.2. Модель учета внесения ошибок в разные модули

В данной модели ПО разделяется на две части. Считается, что 1-я часть ПО содержит N_1 оставшихся ошибок, вторая часть – N_2, общее число оставшихся ошибок ПО $N = N_1 + N_2$. Предполагается, что выявление оставшихся ошибок равновероятно, обнаруживается одна ошибка в заданный интервал времени и исправляется после обнаружения.

Можно показать, что:

$$\begin{cases} N_1(i) = N_1 - i + \sum_{j=0}^{i} \chi_j; \\ N_2(i) = N_2 - \sum_{j=0}^{i} \chi_j, \end{cases}$$

где: χ_j – характеристическая функция, такая что:

$$\chi_j = \begin{cases} 0, \text{ если j-я ошибка найдена в 1-й части ПО;} \\ 1, \text{если j-ая ошибка найдена во 2-ой части ПО.} \end{cases}$$

В этом случае можно рассчитать вероятности обнаружения ошибки на заданном интервале времени тестирования $(t_i, t_{i+1}]$:

$$\begin{cases} p_1(i) = \dfrac{N_1(i)}{N_1(i) + N_2(i)} = \dfrac{N_1 - i + \sum_{j=0}^{i} \chi_j}{N_1 - i + N_2}, \\ p_2(i) = \dfrac{N_2(i)}{N_1(i) + N_2(i)} = \dfrac{N_2 - \sum_{j=0}^{i} \chi_j}{N_1 - i + N_2}. \end{cases}$$

Параметры N_1 и N_2 можно найти, используя метод максимального правдоподобия. Можно показать, что функция максимального правдоподобия имеет следующий вид:

$$L(\chi_1, \chi_2, \ldots, \chi_n) = \prod_{i=1}^{n} (p_1(i-1)^{1-\chi_i} (p_2(i-1)^{\chi_i},$$

где: n — число удаленных оставшихся ошибок.

Нахождение экстремума логарифма функции позволяет получить систему уравнений:

$$\begin{cases} \sum_{i=1}^{n} \dfrac{1 - \chi_i}{N_1 - i + 1 + \sum_{j=0}^{i-1} \chi_j} - \dfrac{1}{N_1 + N_2 - i + 1} = 0; \\ \sum_{i=1}^{n} \dfrac{\chi_i}{N_1 - \sum_{j=0}^{i-1} \chi_j} - \dfrac{1}{N_1 + N_2 - i + 1} = 0. \end{cases}$$

Решение системы уравнений можно найти численными методами.

3.2.3.4. Модель контроля функциональных объектов

Данная модель используется аккредитованной лабораторией «Эшелон» при проведении испытаний на отсутствие недекларированных возможностей [49]. Согласно руководящему документу Гостехкомиссии России [69], в процессе динамического анализа ПО контролируется p заданный процент от $M_{фо}$ идентифицированных функциональных объектов. В процессе статического анализа произвольно выбираются $m_{фо}$ функциональных объектов, в которые вносятся тестовые ошибки. При тестировании фиксируются найденные тестовые ошибки — s и найденные реальные ошибки — n. Используя метод максимального правдоподобия, можно получить оценку числа ошибок в ПО:

$$N = n\frac{M_{фо} - m_{фо}}{\frac{p}{100}M_{фо} - s}.$$

3.2.3.5. Модель испытаний независимыми группами

Данная модель, получившая также название «парная оценка», предполагает проведение тестирования 2-мя независимыми группами тестирования.

В процессе тестирования подсчитывается количество обнаруженных ошибок обеими группами – N_1 и N_2, а также число обнаруженных обеими группами совпавших ошибок – N_{12}.

Обозначив N как число первоначальных ошибок, можно определить эффективность тестирования каждой группы: $E_1 = N_1/N$ и $E_2 = N_2/N$. Гипотетически предполагая одинаковую эффективность тестирования обеих групп, можно допустить, что если одна группа обнаружила определенное количество всех ошибок, то она же могла бы определить то же количество любого случайным образом выбранного подмножества. Это позволяет получить следующие равенства для обеих групп:

$$N_1 \big/ N = N_{12} \big/ N_1 ;$$
$$N_2 \big/ N = N_{12} \big/ N_2 .$$

Считая, что эффективность групп также одинакова между собой, имеем:

$$E_1 = E_2 = N_1 \big/ N = N_{12} \big/ N_2 ,$$

что и дает основное расчётное выражение числа первоначальных ошибок в ПО:

$$N = N_1 \, N_2 \big/ N_{12} .$$

Количество не найденных ошибок будет равно:

$$\bar{N} = N - (N_1 + N_2 - N_{12}).$$

Данная модель полезна на практике, когда тестирование параллельно проводит группа экспертов, имеющих собственные АРМ тестирования, что часто бывает при выездных испытаниях при ограничениях по времени работы.

3.2.4. Модели сложности программного обеспечения

Модели сложности ПО основаны на гипотезе о том, что уровень безошибочности ПО может быть предсказан с помощью показателей (метрик) сложности ПО. Это справедливо для непреднамеренных уязвимостей, так как, чем сложнее и больше программа, тем выше вероятность того, что программист ошибется при ее написании и модификации, а также тем сложнее локализовать ошибку в ПО [77].

В качестве аргументов моделей, как правило, используются метрики[34] сложности ПО, а сами модели сложности можно разделить на аналитические и статистические. Приведем примеры наиболее известных моделей.

3.2.4.1. Метрическая модель ошибок Холстеда

Самой известной аналитической моделью сложности ПО является модель Холстеда [82]. В основу разработки модели положены две базовые характеристики ПО: $\eta = \eta_1 + \eta_2$ – словарь операторов и операндов языка программирования и $N = N_1 + N_2$ – число использования операторов и операндов в программных реализациях, а также гипотеза, что частота использования операторов и операндов в программе пропорциональна двоичному логарифму количества их типов (по аналогии с теорией информации).

Полагаясь на общие статистические физиологические характеристики программиста, реализующего программы, измеряемые указанными метриками, получен ряд эмпирических моделей оценки показателей качества ПО.

Например, сложность программы предложено рассматривать как совокупность интеллектуальных усилий (решения элементарных задач человеком до возникновения ошибки) при кодировании текста на определенном языке программирования:

$$E = \tilde{N} \log_2 \left(\eta / L\right) = \frac{\eta_1 N_2 N \log_2 \eta}{2\eta_2},$$

где: $\tilde{N} = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2$ – теоретическая длина программы; $\eta = \eta_1 + \eta_2$ – число уникальных операторов и операндов языка программирования; $L = 2\eta_2 / (\eta_1 N_2)$ – уровень программы; $N = N_1 + N_2$ – число обращений к операторам и операндам в ПО.

[34] IEEE Std. 1061–1998.

Время кодирования программы рассчитывается следующим образом:

$$\tilde{T} = \frac{E}{S} = \frac{\eta_1 N_2 N \log_2 \eta}{2\eta_2 S} = \frac{\eta_1 N_2 N \log_2 \eta}{36\eta_2},$$

где: $S = \overline{5;20}$ – число Страуда (число мысленных сравнений в секунду), Холстед предложил для мужчины-программиста $S = 18$.

Для расчета B первоначального числа ошибок в ПО предложено использовать выражение:

$$B \approx \frac{E^{\frac{2}{3}}}{3000} \approx \frac{V}{3000} = \frac{N \log_2(\eta)}{3000},$$

где: $V = N \log_2(\eta)$ – объем программы (в бит информации).

3.2.4.2. Многофакторная модель сложности

К простым статистическим моделям сложности можно отнести феноменологическую модель фирмы TRW [78]. Феноменологическая модель представляет собой линейную модель оценки показателя безошибочности ПО по 5-ти значимым характеристикам программ, а именно: уровням логической сложности, сложности взаимосвязей, сложности вычислений, сложности ввода-вывода и понятности.

Феноменологическая модель представляет собой линейную модель оценки показателя сложности ПО по 5-ти эмпирическим характеристикам программ, а именно: логической сложности L_{tot}, сложности взаимосвязей C_{inf}, сложности вычислений C_c, сложности ввода вывода C_{io} и понятности U_{read}.

Показатель логической сложности определяется числом логических операторов:

$$L_{tot} = \frac{z_{ls}}{z_{ex}} + x_{loop} + x_{if} + 0,001 z_{br},$$

где: z_{ls} – число логических операторов; z_{ex} – число исполняемых операторов; x_{loop} – нормированное по вложенности число циклов; x_{if} – нормированное по вложенности число условных операторов; z_{br} – число всех (условных и безусловных) переходов.

Показатель сложности взаимосвязей определяется числом вызываемых программ:

$$C_{inf} = z_{ap} + 0,5 z_{sys},$$

где: z_{ap} – число вызовов прикладных программ; z_{sys} – число вызовов системных программ.

Показатель сложности вычислений определяется числом арифметических операторов:

$$C_c = \frac{z_{cs}}{z_{ex}} \left(\frac{\sum_{ПО} L_{tot}}{\sum_{ПО} z_{cs}}\right) z_{cs},$$

где: z_{cs} – число арифметических операций; z_{ex} – число исполняемых операторов.

Показатель сложности ввода-вывода измеряется числом операций ввода-вывода:

$$C_{io} = \frac{z_{io}}{z_{ex}} \left(\frac{\sum_{ПО} L_{tot}}{\sum_{ПО} z_{io}}\right) z_{io},$$

где: z_{io} – число операторов ввода-вывода; z_{ex} – число исполняемых операторов.

Показатель понятности определяется числом комментариев:

$$U_{read} = z_{com} / (z_{ts} + z_{com}),$$

где: x_{ts} – общее число операторов; z_{com} – число комментариев.

Общий показатель сложности имеет следующий вид:

$$C = L_{tot} + 0,1C_{inf} + 0,2C_c + 0,4C_{io} + (-0,1)U_{read}.$$

Для расчета числа ошибок N предлагается использовать линейную модель:

$$N = L_{tot}\kappa_1 + 0,1C_{inf}\kappa_2 + 0,2C_c\kappa_3 + 0,4C_{io}\kappa_4 + (-0,1)U_{read}\kappa_5,$$

где: κ_i – коэффициент корреляции числа ошибок с i-м показателем сложности.

Значения коэффициентов κ_i легко найти методом наименьших квадратов.

3.2.5. Выбор модели оценки и планирования испытаний

Отметим, что не существует универсальной модели оценки и планирования испытаний ПО. Более того, кроме рассмотренных классов моделей в литературе можно встретить имитационные модели [3,73], структурные [36,66], нечеткие [50], интервальные модели [99], моде-

ли динамической сложности программ [51], модели программно-аппаратных комплексов [76], а также нейронные сети, получившие применение для решения отдельных научных задач. Для выбора подходящих моделей можно предложить ряд качественных и количественных критериев [49].

Качественными критериями можно назвать следующие:

1. Простота использования. В первую очередь касается степени адекватности модели системе сбора статистики, т.е. используемые входные данные могут быть легко получены, они должны быть представительны, входные и выходные данные понятны экспертам.

2. Достоверность, т.е. модель должна обладать разумной точностью, необходимой для решения задач анализа или синтеза в области безопасности ПО. Положительным свойством модели является возможность использования априорной информации и комплексирования данных других моделей с целью сокращения входной выборки.

3. Применимость для решения различных задач. Некоторые модели позволяют получить оценки широкого спектра показателей, необходимых экспертам на различных этапах жизненного цикла ПО, например, показатели надежности, ожидаемое число ошибок различных типов, прогнозируемые временные и стоимостные затраты, квалификацию специалистов, качество тестов, показатели точности, показатели покрытия ПО и др.

4. Простота реализации, в том числе, возможность автоматизируемости процесса оценки на базе известных математических пакетов и библиотек, переобучения модели после доработок, учета случаев неполных или некорректных входных статистик, учета других ограничений моделей.

В качестве количественных критериев используют следующие:

— показатели точности оценки;

— показатели качества прогнозирующих моделей (сходимость, устойчивость к шуму, точность предсказания, согласованность);

— информационные критерии качества прогнозирующих моделей (размерность, критерии BIC/AIC).

— комбинированные и интегральные показатели, например:

$$IC = \max \sum_{i=1}^{K} k_i \chi_i,$$

где: k_i — весовой коэффициент i-го свойства рассматриваемой модели, выбираемой экспертом; χ_i — характеристическая функция i-го свойства.

Исследование показало, что существует весьма большое количество математических моделей, позволяющих получить оценки показателей технологической безопасности ПО на различных этапах жизненного цикла, что важно при планировании затрат на информационную безопасность. Рассмотренная классификация моделей позволяет на практике сориентироваться при выборе и комплексировании моделей в зависимости от имеющейся статистики.

Надо понимать, что в виду динамичности, сложности и разнородности современных программных проектов, к указанным моделям не могут предъявляться высокие требования по точности, и они носят зачастую интуитивный характер при принятии решений по планированию тестирования ПО на всем множестве входных данных. Несмотря на это, результаты применения моделей удобно использовать как при обосновании трудоемкости испытаний, так и при предоставлении отчетных материалов, что повышает уверенность заказчика в результатах выполненных работ.

3.3. Модели управления доступом

Основной подсистемой комплексного СЗИ является подсистема управления доступом. В руководящих документах Гостехкомиссии России определены два принципа управления доступом: дискреционный и мандатный. В ГОСТ ИСО/МЭК 15408, кроме указанных, описан еще ролевой принцип управления доступом, однако допускаются любые другие недискреционные принципы управления доступом [10, 45, 81].

Формальные модели управления доступом используются в тех случаях, когда те или иные утверждения о стойкости систем защиты информации требуют строгого доказательства — например, в случае, если оценочный стандарт требует использования формальных методов проектирования средств защиты информации[35]. На практике применение таких методов является чрезвычайно трудоемкой и дорогостоящей задачей, поэтому их практическое использование весьма и весьма ограничено [84].

На сегодняшний день используются четыре основных класса моделей управления доступом:

1. Дискреционные модели управления доступом — модели, в которых владелец ресурса сам задает права доступа к нему. В большинстве

[35] ГОСТ Р ИСО/МЭК 15408–3-2008

случаев права доступа субъектов к объектам представляются в виде матрицы или списков доступа.

2. Мандатные модели управления доступом, в которых режим доступа субъектов к объектам определяется установленным режимом конфиденциальности.

3. Ролевая модель управления доступом, копирующая иерархическую структуру организации и позволяющая упростить администрирование.

4. Атрибутная модель, являющаяся наиболее универсальной и позволяющая контролировать доступ с учетом произвольных параметров среды, субъектов и объектов доступа.

3.3.1. Дискреционная модель управления доступом

Классической дискреционной моделью управления доступом является модель Харрисона-Руззо-Ульмана (Harrison-Ruzzo-Ullman model), которая формализует понятие матрицы доступа — таблицы, описывающей права доступа субъектов к объектам (рис. 3.3).

Строки матрицы доступа соответствуют субъектам, существующим в системе, а столбцы — объектам. На пересечении строки и столбца указаны права доступа соответствующего субъекта к данному объекту: например, на рис. 3.3 субъект *subj 3* обладает правами чтения и записи по отношению к объекту *obj 3*.

Введём следующие *обозначения*:

— S — множество возможных субъектов;

— O — множество возможных объектов (напомним, что $S \subset O$);

— $R = \{r_1, ..., r_n\}$ — конечное множество прав доступа;

— $O \times S \times R$ — пространство состояний системы;

— M — матрица прав доступа, описывающая текущие права доступа субъектов к объектам;

— $Q = (S, O, M)$ — текущее состояние системы;

— $M[s, o]$ — ячейка матрицы, содержащая набор прав доступа субъекта $s \in S$ к объекту $o \in O$.

	obj 1	obj 2	obj 3	...
subj 1	r	...		
subj 2				
subj 3	...		r, w	
...				

Рис. 3.3. Матрица доступа

Поведение системы во времени моделируется переходами между различными её состояниями. Переходы осуществляются путём внесения изменений в матрицу M с использованием *команд* следующего вида:

command $\alpha(x_{1,\ldots,}x_k)$
if r_1 in $M[x_{s1}, x_{o1}]$ and r_2 in $M[x_{s2}, x_{o2}]$ and ... r_m in $M[x_{sm}, x_{om}]$
then
op_1,
op_2,
...
op_n,
end

Здесь α — имя команды; x_i — параметры команды, представляющие собой идентификаторы субъектов и объектов, op_i — элементарные операции.

Элементарные операции $op_1 \ldots op_n$ будут выполнены в том случае, если выполняются все без исключения условия из блока if ... then.

При описании элементарных операций мы будем полагать, что в результате выполнения операции система переходит из состояния $Q = (S, O, M)$ в состояние $Q' = (S', O', M')$.

Модель предусматривает наличие шести элементарных операций:

1. enter r into $M[s, o]$ ($s \in S, o \in O$) — добавление субъекту s права r по отношению к объекту o. В результате выполнения команды происходят следующие изменения в состоянии системы:
   - $S' = S$,
   - $O' = O$,
   - $M'[x_s, x_o] = M[x_s, x_0]$, *если* $(x_s, x_o) \neq (s, o)$,
   - $M'[s, o] = M[s, o] \cup \{r\}$.

Заметим, что содержимое ячейки таблицы рассматривается как множество. Это, в частности, означает, что если добавляемый элемент уже присутствовал в ячейке, то её содержимое не изменяется.

2. delete r from $M[s, o]$ ($s \in S, o \in O$) — удаление у субъекта s права r по отношению к объекту o. Изменения в состоянии системы:
   - $S» = S$,
   - $O» = O$,
   - $M» [x_S, x_0] = M [x_S, x_0]$, *если* $(x_S, x_0) \neq (s, o)$,
   - $M» [s, o] = M [s, o] \setminus \{r\}$.

Если удаляемое право отсутствовало в ячейке, то состояние системы в результате выполнения данной команды никак не изменяется.

3. create subject s ($s \notin S$) — создание нового субъекта s. Изменения в состоянии системы:

$O' = O \cup \{s\}$,

$S' = S \cup \{s\}$,

$M'[x_s, x_o] = M[x_s, x_o]$ *для* $\forall (x_s, x_o) \in S \times O$,

$M'[s, x_o] = \emptyset$ *для* $\forall x_o \in O'$

$M'[s, x_s] = \emptyset$ *для* $\forall x_s \in S'$

Как видим, при создании субъекта в матрицу M добавляются строка и столбец.

4. destroy subject s ($s \in S$) — удаление существующего субъекта s.

Изменения в состоянии системы:

— $S' = S \setminus \{s\}$,

— $O' = O \setminus \{s\}$,

— $M'[x_s, x_o] = M[x_s, x_o]$ *для* $\forall (x_s, x_o) \in S' \times O'$.

5. create object o ($o \notin O$) — создание нового объекта o.

Изменения в состоянии системы:

— $O' = O \cup \{o\}$,

— $S' = S$,

— $M'[x_s, x_o] = M[x_s, x_o]$ *для* $\forall (x_s, x_o) \in S \times O$,

— $M'[x_s, o] = \emptyset$ *для* $\forall x_s \in S'$

При добавлении объекта в матрице доступа создаётся новый столбец.

6. destroy object o ($o \in O \setminus S$) — удаление существующего объекта o.

Изменения в состоянии системы:

— $O' = O \setminus \{o\}$,

— $S' = S$,

— $M'[x_s, x_o] = M[x_s, x_o]$ *для* $\forall (x_s, x_o) \in S' \times O'$.

Формальное описание системы в модели Харрисона-Руззо-Ульмана выглядит следующим образом. *Система* $\Sigma = (Q, R, C)$ состоит из следующих элементов:

1. Конечный набор прав доступа $R = \{r_1, ..., r_n\}$.
2. Конечный набор исходных субъектов $S_0 = \{s_1, ..., s_l\}$.
3. Конечный набор исходных объектов $O_0 = \{o_1, .., o_m\}$.
4. Исходная матрица доступа M_0.
5. Конечный набор команд $C = \{\alpha_i(x_1, ..., x_k)\}$.

Поведение системы во времени рассматривается как последовательность состояний $\{Q\}$, каждое последующее состояние является результатом применения некоторой команды к предыдущему: $Q_{n+1} = C_n(Q_n)$.

Для заданной системы начальное состояние $Q_0 = \{S_0, O_0, M_0\}$ называется *безопасным относительно права r*, если не существует приме-

нимой к Q_0 последовательности команд, в результате выполнения которой право r будет занесено в ячейку матрицы M, в которой оно отсутствовало в состоянии Q_0. Другими словами это означает, что субъект никогда не получит право доступа r к объекту, если он не имел его изначально.

Если же право r оказалось в ячейке матрицы M, в которой оно изначально отсутствовало, то говорят, что произошла *утечка права r*.

С практической точки зрения значительный интерес представлял бы универсальный метод определения того, является ли заданная система с некоторым начальным состоянием безопасной относительно того или иного права доступа. Покажем, как эта задача может быть решена для одного из частных случаев.

Система $\Sigma = (Q, R, C)$ называется *монооперационной*, если каждая команда $\alpha_i \in C$ выполняет один примитивный оператор.

Теорема. *Существует алгоритм, который проверяет, является ли исходное состояние монооперационной системы безопасным для данного права a.*

К сожалению, расширить полученный результат на произвольные системы невозможно.

Теорема. *Для систем общего вида задача определения того, является ли исходное состояние системы безопасным для данного права a, является вычислительно неразрешимой.*

Классическая модель Харриона-Руззо-Ульмана до сих пор широко используется при проведении формальной верификации корректности построения систем разграничения доступа в высоко защищённых автоматизированных системах. Развитие моделей дискреционного управления доступом заключается преимущественно в построении всевозможных модификаций модели Харрисона-Руззо-Ульмана, а также в поиске минимально возможных ограничений, которые можно наложить на описание системы, чтобы вопрос её безопасности был вычислительно разрешимым.

3.3.2. Мандатная модель управления доступом

Одной из самых известных моделей мандатного управления доступа является модель Белла-ЛаПадулы (Bell-LaPadula Model). Мандатный принцип разграничения доступа, изначально, ставил своей целью перенести на автоматизированные системы практику секретного документооборота, принятую в правительственных и военных структурах, когда все документы и допущенные к ним лица ассоциируются с иерархическими уровнями секретности.

В модели Белла-ЛаПадулы по грифам секретности распределяются субъекты и объекты, действующие в системе, и при этом выполняются следующие правила:

1. *Простое правило безопасности* (*Simple Security, SS*).

Субъект с уровнем секретности x_s может читать информацию из объекта с уровнем секретности x_o тогда и только тогда, когда x_s преобладает над xo.

2. **-свойство* (**-property*).

Субъект с уровнем секретности x_s может писать информацию в объект с уровнем секретности x_o в том и только в том случае, когда x_o преобладает над x_s.

Для первого правила существует мнемоническое обозначение *No Read Up*, а для второго – *No Write Down*.

Диаграмма информационных потоков, соответствующая реализации модели Белла-ЛаПадулы в системе с двумя уровнями секретности, приведена на рис. 3.4.

Перейдём к формальному описанию системы. Введём следующие обозначения:

– S – множество субъектов;

– O – множество объектов, $S \subset O$;

– $R = \{r, w\}$ – множество прав доступа, r – доступ на чтение, w – доступ на запись;

– $L = \{U, SU, C, S, TS\}$ – множество уровней секретности, *U-Unclassified, SU – Sensitive but unclassified, C–Confidential, S – Secret, TS – Top secret;*

– $\Lambda = (L, \leq, \bullet, \otimes)$ – решётка уровней секретности;

– V – множество состояний системы, представляемое в виде набора упорядоченных пар (F, M), где:

– $F : S \cup O \rightarrow L$ – функция уровней секретности, ставящая в соответствие каждому объекту и субъекту в системе определённый уровень секретности;

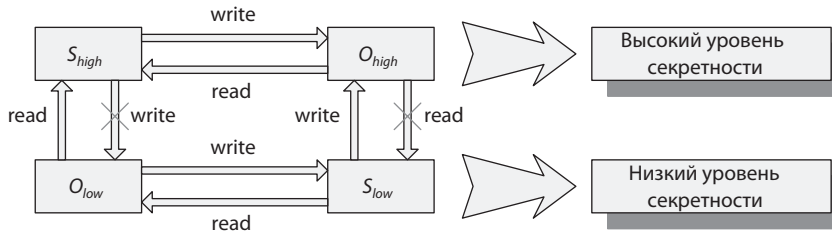

Рис. 3.4. Диаграмма информационных потоков по модели Белла-ЛаПадулы

– M – матрица текущих прав доступа.

Система $\Sigma = (v_0, R, T)$ в модели Белла-ЛаПадулы состоит из следующих элементов:

– v_0 – начальное состояние системы;

– R – множество прав доступа;

– $T : V \times R \to V$ – функция перехода, которая в ходе выполнения запросов переводит систему из одного состояния в другое.

Изменение состояний системы во времени происходит следующим образом: система, находящаяся в состоянии $v \in V$, получает запрос на доступ $r \in R$ и переходит в состояние $v^* = T(v, r)$.

Состояние v_n называется *достижимым* в системе $\Sigma = (v_0, R, T)$, если существует последовательность $\{(r_0, v_0), \ldots, (r_{n-1}, v_{n-1}), (r_n, v_n)\}$: : $T(r_i, v_i) = v_{i+1} \forall i = 0, n-1$. Начальное состояние v_0 является достижимым по определению.

Состояние системы (*F, M*) называется *безопасным по чтению* (или *simple-безопасным*), если для каждого субъекта, осуществляющего в этом состоянии доступ по чтению к объекту, уровень безопасности субъекта доминирует над уровнем безопасности объекта:

$$\forall s \in S, \forall o \in O, \quad r \in M[s, o] \to F(o) \le F(s).$$

Состояние (*F, M*) называется *безопасным по записи* (или ** – безопасным*) в случае, если для каждого субъекта, осуществляющего в этом состоянии доступ по записи к объекту, уровень безопасности объекта доминирует над уровнем безопасности субъекта:

$$\forall s \in S, o \in O : \quad w \in M[s, o] \to F(s) \le F(o).$$

Состояние (*F, M*) называется *безопасным*, если оно безопасно по чтению и по записи.

Наконец, система $\Sigma = (v_0, R, T)$ называется *безопасной*, если её начальное состояние v_0 безопасно, и все состояния, достижимые из v_0 путём применения конечной последовательности запросов из R, безопасны.

Теорема (*Основная теорема безопасности Белла-ЛаПадулы*). *Система* $\Sigma = (v_0, R, T)$ *безопасна тогда и только тогда, когда выполнены следующие условия:*

1. Начальное состояние v_0 *безопасно.*

2. Для любого состояния v, *достижимого из* v_0 *путём применения конечной последовательности запросов из* R, *таких, что* $T(v, r) = v^*$, $v = (F, M)$ *и* $v^* = (F^*, M^*)$, *для* $\forall s \in S, \forall o \in O$ *выполнены условия:*

1. Если $r \in M^*[s, o]$ *и* $r \notin M[s, o]$, *то* $F^*(o) \le F^*(s)$.

2. *Если* $r \in M[s,o]$ *и* $F^*(s) < F^*(o)$, *то* $r \notin M^*[s,o]$.
3. *Если* $w \in M^*[s,o]$ *и* $w \notin M[s,o]$, *то* $F^*(s) \leq F^*(o)$.
4. *Если* $w \in M[s,o]$ *и* $F^*(o) < F^*(s)$, *то* $w \notin M^*[s,o]$.

Отметим, что изложенная модель в силу своей простоты имеет целый ряд серьёзных недостатков. Например, никак не ограничивается вид функции перехода T — а это означает, что можно построить функцию, которая при попытке запроса на чтения к объекту более высокого уровня секретности до проверки всех правил будет понижать уровень секретности объекта. Другим принципиальным недостатком модели Белла-ЛаПадулы является потенциальная возможность организации скрытых каналов передачи информации [92]. Тем самым, дальнейшее развитие моделей мандатного управления доступом было связано с поиском условий и ограничений, повышающих её безопасность.

В настоящее время модель Белла-ЛаПадулы и другие модели мандатного управления доступом широко используются при построении и верификации АС, преимущественно предназначенных для работы с информацией, составляющей государственную тайну.

3.3.3. Ролевая модель управления доступом

Ролевая модель управления доступом содержит ряд особенностей, которые не позволяют отнести её ни к категории дискреционных, ни к категории мандатных моделей. Основная идея реализуемого в данной модели подхода состоит в том, что понятие «субъект» заменяется двумя новыми понятиями:

– *пользователь* — человек, работающий в системе;

– *роль* — активно действующая в системе абстрактная сущность, с которой связан ограниченный и логически непротиворечивый набор полномочий, необходимых для осуществления тех или иных действий в системе.

Классическим примером роли является root в Unix-подобных системах — суперпользователь, обладающий неограниченными полномочиями. Данная роль по мере необходимости может быть задействована различными администраторами.

Основным достоинством ролевой модели является близость к реальной жизни: роли, действующие в АС, могут быть выстроены в полном соответствии с корпоративной иерархией и при этом привязаны не к конкретным пользователям, а к должностям — что, в частности, упрощает администрирование в условиях большой текучки кадров.

Управление доступом при использовании ролевой модели осуществляется следующим образом:

1. Для каждой роли указывается набор полномочий, представляющий собой набор прав доступа к объектам АС.
2. Каждому пользователю назначается список доступных ему ролей.

Отметим, что пользователь может быть ассоциирован с несколькими ролями — данная возможность также значительно упрощает администрирование сложных корпоративных АС.

Перейдём к формальному описанию системы. Введём следующие обозначения:

– U — множество пользователей;
– R — множество ролей;
– P — совокупность полномочий на доступ к объектам (реализованная, например, в виде матрицы доступа);
– S — множество сеансов работы пользователей с системой

Управление доступом реализуется с использованием следующих отображений:

– $PA \subseteq P \times R$ — отображение множества полномочий на множество ролей, задающее для каждой роли установленный набор полномочий;
– $UA \subseteq U \times R$ — отображение множества пользователей на множество ролей, определяющее набор ролей, доступных данному пользователю;
– $user : S \to U$ — функция, определяющая для сеанса $s \in S$ текущего пользователя $u \in U$:

$user(s) = u$;

$roles : S \to \{R\}$ — функция, определяющая для сеанса $s \in S$ набор ролей из множества R, доступных в данном сеансе:

– $roles(s) = \{r_i \mid (user(s), r_i) \in UA\}$;

$permissions : S \to \{P\}$ — функция, задающая для сеанса $s \in S$ набор доступных в нём полномочий (иначе говоря, совокупность полномочий всех ролей, доступных в данном сеансе):

$$permissions(s) = \bigcup_{r \in roles(s)} \{p_i \mid (p_i, r) \in PA\}.$$

Взаимосвязь пользователей, ролей, полномочий и сеансов показана на рис. 3.5.

Критерий безопасности системы при использовании ролевой модели звучит следующим образом: система считается *безопасной*, если

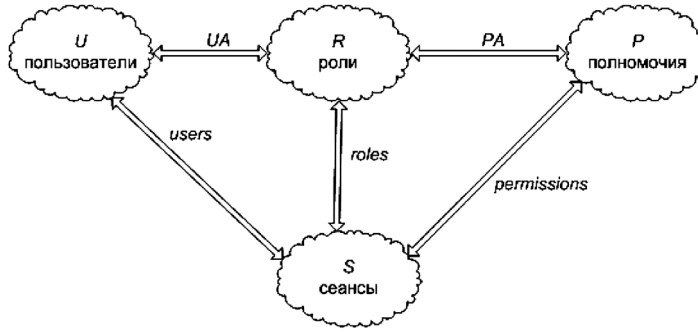

Рис. 3.5. Взаимосвязь ролей, полномочий, пользователей и сеансов

любой пользователь в системе, работающий в сеансе $s \in S$, может осуществлять действия, требующие полномочий $p \in P$, только в том случае, если $p \in permissions(s)$.

На практике управление доступом в АС при использовании ролевой модели осуществляется главным образом не с помощью назначения новых полномочий ролям, а путём задания отношения UA — т.е. путём определения ролей, доступных данному пользователю.

3.3.4. Атрибутная модель управления доступом

Атрибутная модель управления доступом является наиболее универсальной. Основная идея данной модели заключается в том, что решение о предоставлении доступа субъекта к объекту принимается на основе анализа набора произвольных атрибутов, связанных с субъектом, объектом или даже средой их функционирования. Каждый атрибут представляет собой некий набор данных, который может быть сопоставлен с массивом допустимых значений данного атрибута.

Для получения доступа к объекту субъект предъявляет некий имеющийся у него набор атрибутов. Монитор безопасности обращений сравнивает значение каждого атрибута с перечнем допустимых и в случае, если все атрибуты являются допустимыми, предоставляет доступ.

Формальное описание модели выглядит следующим образом. Введем ряд обозначений:

- S — множество субъектов;
- O — множество объектов;
- E — множество элементов среды;
- $SA_k, k = \overline{1, K}$ — множество возможных атрибутов субъектов;

– $OA_m, m = \overline{1, M}$ – множество возможных атрибутов объектов;

– $EA_n, n = \overline{1, N}$ – множество возможных атрибутов элементов среды.

Для задания атрибутов конкретным субъектам, объектам или элементам среды используются следующие три отношения:

$$ATTR(s) \subseteq SA_1 \times SA_2 \times \ldots \times SA_K;$$

$$ATTR(o) \subseteq OA_1 \times OA_2 \times \ldots \times OA_M;$$

$$ATTR(e) \subseteq EA_1 \times EA_2 \times \ldots \times EA_M.$$

В общем случае правило политики безопасности, разрешающее или запрещающее доступ, представляет собой булеву функцию следующего вида:

$$Rule\ X: can\_access\ (s, o, e) \leftarrow f\,(ATTR\,(s), ATTR\,(o), ATTR\,(e))$$

Принципиально важным преимуществом данной модели является тот факт, что субъект доступа совершенно не обязательно должен быть заранее зарегистрирован в системе: необходимым является исключительно наличие у него соответствующих атрибутов.

В заключении, следует сказать, что можно выделить два основных подхода к использованию формальных моделей управления доступом непосредственно при проведении оценки соответствия средств защиты информации:

1. Независимый контроль факта реализации той или иной модели в системе, а также контроль выполнения формального критерия безопасности (если применимо).

2. Формальное доказательство тех или иных положений с использованием инструментария той или иной модели.

Первый подход, в частности, характерен для ролевой модели или модели Белла-ЛаПадулы, а второй – для модели Харрисона-Руззо-Ульмана.

3.4. Метрики парольных систем

По статистике парольная подсистема считается самой уязвимой в современных защищенных АС, поэтому требует самого внимательного отношения при оценке соответствия.

Пароль – идентификатор субъекта доступа, который является его (субъекта) секретом [69]. Согласно существующей статистике наиболее популярными паролями пользователей являются «123456» и «qwerty». Это определяет необходимость формирования критериев стойкости парольной защиты и методов оценки выполнения установленных критериев. На сегодняшний момент существует разнообразное количество рекомендаций по выбору паролей как неофициальных подходов, так и закрепленных на законодательном уровне [36].

Введем определения, используемые при дальнейшем изложении. Пусть A – алфавит паролей, обозначим пароль длины n как последовательность символов: $\tilde{p}=\left(p_1, p_2, \ldots, p_n\right) \in A^*, p_i \in A$. Таким образом, число всевозможных паролей длины n, которые можно составить из символов алфавита A, составляет $\left|A\right|^n$.

Среди основных метрик парольной защиты могут быть выделены следующие:

– длина пароля n: большинство рекомендаций устанавливают минимальную длину пароля равной 8 символам;

– мощность алфавита пароля $|A|$: например, расширение алфавита паролей специальными символами или буквами в верхнем регистре повышает стойкость парольной системы;

– срок действия пароля T: большинство документов рекомендует использовать пароли временного действия, что позволяет повысить стойкость парольной защиты.

Пусть V – скорость подбора пароля злоумышленником, тогда вероятность P_G подбора пароля в течение его срока действия может быть выражена следующим образом:

$$\mathbb{P}_G = \frac{V \cdot T}{\left|A\right|^n}.$$

Обычно скорость подбора паролей V и срок действия пароля T можно считать известными. Задав допустимое значение вероятности $\mathbb{P}_G$ подбора пароля в течение его срока действия, можно определить требуемую мощность пространства паролей $\left|A\right|^n$.

[36] РД «Автоматизированные системы. Защита от несанкционированного доступа к информации. Классификация автоматизированных систем и требования по защите информации» (Гостехкомиссия России, 1992).

В случае использования генераторов псевдослучайной последовательности при формировании паролей, сложность пароля можно оценить с использованием понятия *энтропии*. Понятие информационной энтропии было впервые формализовано Шенноном как мера неопределенности или непредсказуемости информации, неопределённость появления какого-либо символа алфавита. Энтропия множества $U=\{u_1, u_2, \ldots, u_N\}$ оценивается с использованием следующего выражения:

$$H(U)=-\sum_{i=1}^{N} p_i \cdot \log_2 p_i,$$

где: p_i– вероятность появления элемента u_i. Если все события появлений элементов равновероятны (т.е. $p_i = 1/N$ для $\forall i \in [1, N]$), то выражение принимает вид:

$$H(U)=-\sum_{i=1}^{N} p_i \cdot \log_2 p_i=-\sum_{i=1}^{N} \frac{1}{N} \cdot \log_2 \frac{1}{N}= \log_2 N.$$

В случае $U= A^*$ информационная энтропия парольной системы определяется по формуле:

$$H(A^*)= \log_2 |A^*|= \log_2 |A|^n = n \cdot \log_2 |A|.$$

Величина $H(A^*)$ характеризует степень случайности пароля при его генерации и показывает, насколько сложно его угадать злоумышленнику. Например, для известного пароля энтропия равна нулю. Если пароль имеет энтропию равную 1 символу, то угадать его с первой попытки можно с вероятностью равной $1/|A|$. В таблице 3.11 приведены значения энтропии в расчете на один символ для различных алфавитов.

Для оценки стойкости парольных систем, в которых используются генераторы псевдослучайной последовательности, могут быть использованы различные статистические тесты. Генератор псевдослучайной последовательности рассматривается как программа, которая генерирует битовые последовательности вида $\tilde{s}=(s_1, s_2, \ldots, s_n), s_i \in \{0,1\}$. Данные последовательности подвергают-

Таблица 3.11

Значения энтропии в расчете на один символ для различных алфавитов

Алфавит *A*	**Число символов** *N*	**Энтропия на один символ** $\log_2\|A\|$
Арабские цифры (0—9)	10	3,3219
Шестнадцатеричные числа (0—9, A-F)	16	4,0
Латинский алфавит (a-z, A-Z)	52	5,7004
Латинский алфавит с цифрами (a-z, A-Z, 0—9)	62	5,9542
Таблица ASCII	94	6,5546

ся различным статистическим тестам, в ходе которых определяется мера их случайности. Методы тестирования описываются с использованием нулевой гипотезы H_0 (тестируемая последовательность является случайной) и альтернативной гипотезы H_1 (тестируемая последовательность не является случайной). Тест разрабатывается с целью проверки нулевой гипотезы H_0.

Процесс статистического тестирования в общем случае выглядит следующим образом.

1. Формулируется гипотеза H_0 — последовательность $\tilde{s}$ является случайной с равномерным распределением — и противоположная ей гипотеза H_a.

2. Фиксируется уровень значимости a.

3. Задается статистика $T:\{0,1\}^n \to \mathbb{R}$.

4. Для заданной последовательности $\tilde{s}$ вычисляется статистика: $T=T(\tilde{s})$.

5. Вычисляется достигаемый уровень значимости $p=p(T)$ для полученного значения статистики.

6. Если $p \geq \alpha$, принимается нулевая гипотеза H_0; иначе гипотеза H_0 отвергается.

Назовем *периодом последовательности* $\tilde{s}=(s_1,s_2,\ldots,s_n)$ такое минимальное число p, что $s_{i+p}=s_i$ для $\forall i>0$. *Серией* длины k, начинающейся с символа t, назовем подпоследовательность такую, что $s_{t-1} \neq s_t=s_{t+1}=\ldots=s_{t+k-1} \neq s_{t+k}$. Соломон Голомб предложил следующие необходимые условия случайности для бинарной последовательности:

1. В каждом периоде количество «1» и «0» должно различаться не более чем на единицу.

2. В каждом периоде $1/2^k$ числа серий должно иметь длину k: половина серий должна иметь длину один, четверть — длину два, восьмая часть — длину три и т. д.

3. Функция автокорреляции должна принимать только два различных значения для всех возможных сдвигов.

Приведем примеры статистических тестов[37].

1. Частотный тест для битов. В ходе проведения данного теста исследуется количество нулей и единиц в последовательности. Тест оценивает приближение числа единиц последовательности к значению $n/2$.

2. Частотный тест для блоков. Цель теста — определить, приближается ли количество единиц в любом блоке длины m к значению $m/2$.

3. Тест серий. Цель теста — сравнить распределение серий в тестируемой последовательности с ожидаемым распределением таких серий для случайной последовательности.

4. Матрично-ранговый тест. Цель теста — проверка линейной зависимости между подстроками фиксированного размера — матрицами 32×32 бита.

Существуют и другие наборы тестов для исследования статистических свойств последовательностей: набор тестов Д. Кнута, набор тестов CRYPT-X, набор тестов DIEHARD и др. Следует отметить, что ни один тест не может гарантировать «случайность». Тем не менее, тесты позволяют выявлять «неслучайные» последовательности.

3.5. Модели периодического инспекционного контроля

Правилами обязательной сертификации СЗИ, устанавливаемыми в нормативных и методических документах регуляторов, предусмотрен периодический инспекционный контроль за стабильностью характеристик СЗИ. Периодичность и объемы испытаний в рамках инспекционного контроля сертифицированных СЗИ определяются в нормативных и методических документах по их сертификации. На практике после получения сертификата соответствия формируется календарный план инспекционного контроля за стабильностью ха-

[37] NIST SP 800–22: 2010.

рактеристик сертифицированных СЗИ, согласно которому проверки проводятся через детерминированные промежутки времени.

По причине необходимости выделения дополнительных средств на испытания СЗИ после сертификации, число моментов контроля является заранее обоснованной конечной величиной. Однако в реальной жизни по ряду случайных факторов сложно организовать инспекционный контроль через строго заданные промежутки времени. Поэтому, представляет интерес применение моделей инспекционного контроля, в которых период контроля может быть как детерминированным, так и стохастическим, но при заданном числе моментов контроля.

В широком смысле инспекционный контроль представляет собой систематическое наблюдение за деятельностью по оценке соответствия как основы поддержания правомерности сертификата соответствия. При этом согласно ГОСТ ИСО/МЭК 15408 контроль касается не только СЗИ (объекта оценки), но и среды функционирования. В таком случае инспекционный контроль может включать как процедуры контроля характеристик СЗИ, так и процедуры контроля характеристик среды функционирования. Рассмотрим стохастические и детерминированные модели указанных процедур.

3.5.1. Модели инспекционного контроля средств защиты информации

Рассмотрим t период жизненного цикла СЗИ с учетом проводимого инспекционного контроля за стабильностью характеристик. Поскольку период времени t во много раз превышает время контроля, положим последнее мгновенным. Тогда степень стабильности характеристик СЗИ характеризуется вероятностью $P\left(\hat{z} < Q\right) = F_{\hat{z}}(Q)$ того, что время $\hat{z}$ обнаружения нарушения (уязвимости, ошибки) в межконтрольном интервале T не превосходит допустимое время Q жизненного цикла СЗИ при наличии нарушения. Фрагмент инспекционного контроля представлен на рис. 3.6.

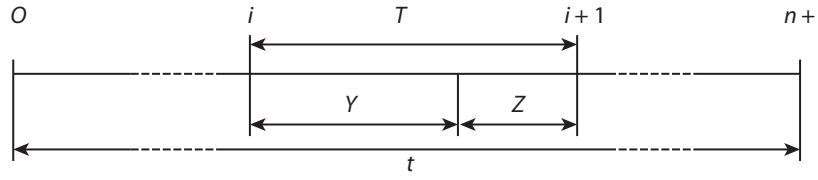

Рис. 3.6. Фрагмент инспекционного контроля средства защиты информации

Будем считать поток нарушений (ошибок, уязвимостей) простейшим с плотностью распределения интервала $\hat{y}$ между ними:

$$g_{\hat{y}} = \lambda e^{-\lambda y},$$

где: λ — интенсивность нарушений.

Зададим стохастическую модель обнаружения нарушений. В этом случае контроль производится определенное число раз с равной вероятностью независимо друг от друга. Таким образом, образованный всеми моментами контроля ограниченный поток является потоком Бернулли с плотностью распределения интервала $\hat{T}$ между моментами контроля [52, 97]:

$$f_{\hat{T}} = n / t(1 - T / t)^{(n-1)},$$

где: n — число моментов контроля.

Величина задержки $\hat{z} = \hat{T} - \hat{y}$ является функцией двух случайных величин и имеет следующую функцию распределения:

$$F_{\hat{Z}}^{s} = \iint_{(s)} n / t(1 - T / t)^{n-1} \lambda e^{-\lambda y} dT dy .$$

Определив пределы интегрирования (рис. 3.7), получим:

$$F_{\hat{z}}^{s} = \int_{0}^{t-z} \left(\int_{y}^{y+z} n / t(1 - T / t)^{n-1} \lambda e^{-\lambda y} dT \right) dy + \\ + \int_{t-z}^{t} \left(\int_{y}^{t} n / t(1 - T / t)^{n-1} \lambda e^{-\lambda y} dT \right) dy.$$

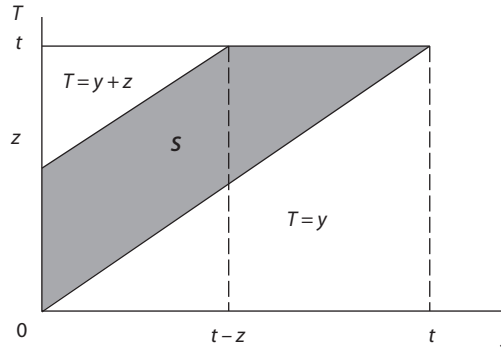

Рис 3.7. Область интегрирования интервала задержки времени обнаружения нарушения

После упрощения имеем следующее выражение:

$$F_{\hat{z}}^{s} = \lambda / t^{n} \ (\int_{0}^{t-z} e^{-\lambda y} \left(\left(t-y\right)^{n} - \left(t-z-y\right)^{n} \right) dy + \int_{t-z}^{t} e^{-\lambda y} \left((t-y)^{n} \right) dy).$$

Разложив в степенной ряд подынтегральные выражения формулы, получим приближенное значение функции распределения, которое и является основным расчетным соотношением:

$$F_{\hat{z}}^{s} = \lambda \sum_{i=0}^{n} \sum_{j=0}^{r} \sum_{l=1}^{n+j+1} (-1)^{i+j+l+1} C_{n}^{i} C_{n+j+1}^{l} \frac{\lambda^{j} t^{j+1-l} z^{l}}{j!(1+j+i)},$$

где: r — число итераций.

Для сравнения стохастической модели с детерминированной рассмотрим последнюю подробнее. В детерминированной модели моменты контроля образуют регулярный поток с постоянной величиной интервала $T = t/(n+1)$ и плотностью распределения времени обнаружения нарушения:

$$g_{\hat{z}} = \lambda e^{-\lambda(T-z)}; \ 0 < z < T.$$

Можно показать, что выражение для функции распределения в детерминированной модели имеет следующий вид:

$$F_{\hat{z}}^{d} = e^{-\lambda T}(e^{\lambda z} - 1); \ 0 < z < T.$$

Сравнивая выражения моделей, можно говорить о соответствии рассмотренных моделей процессу выявления нарушений стабильности характеристик СЗИ (при заданных значениях λ, Q, t и n):

$$F_{\hat{z}}^{s}(z) \leqslant F_{\hat{z}}^{d}(z).$$

Вероятность P_{n} обнаружения нарушений за t период жизненного цикла СЗИ можно представить следующим образом:

$$P_{n} = (n+1) \cdot F_{\hat{z}},$$

где: n — число моментов инспекционного контроля;

$F_{\hat{z}} = max\left(F_{\hat{z}}^{s}\left(z\right), F_{\hat{z}}^{d}(z)\right)$.

Анализ рассмотренных моделей показал преимущество стохастической модели при небольшом числе моментов инспекционного контроля. Понятийно это может быть объяснено тем, что даже при малом числе случайных моментов контроля характеристик СЗИ всегда существует вероятность того, что обнаружение нарушения будет произведено сразу же при его возникновении, в то время как при использовании детерминированной модели период проверки не может быть меньше заданной величины.

3.5.2. Модели инспекционного контроля сред функционирования

Контроль ограничений, предъявляемых к СЗИ, в первую очередь касается проверки среды и условий функционирования и производства СЗИ. Такие проверки позволяют исключить появление нарушений (ошибок, уязвимостей), касающихся внешнего интерфейса СЗИ. В этом смысле процедуры обнаружения нарушений среды можно интерпретировать как механизм предупреждения нарушений СЗИ.

При разработке моделей контроля среды будем придерживаться подхода, приведенного в предыдущем разделе. Будем считать, что степень стабильности характеристик СЗИ характеризуется вероятностью $P\left(\hat{z} < Q\right) = F_{\hat{z}}(Q)$ того, что время предварительного контроля $\hat{z}$ между моментом контроля среды и возможным моментом наступления нарушения характеристик СЗИ не превосходит допустимое время Q. Зададим стохастическую модель контроля нарушений среды (рис. 3.8).

Можно показать, что величина времени предварительного контроля является функцией двух случайных величин $\hat{z} = \hat{y} - \hat{T}$ и имеет следующую функцию распределения:

$$F_{\hat{z}}^{s} = \iint_{(s)} n / t(1 - T / t)^{n-1} \lambda e^{-\lambda y} dT dy,$$

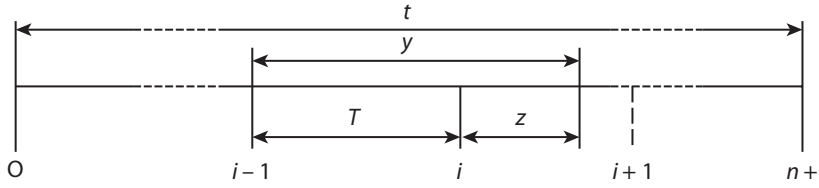

Рис 3.8. Диаграмма функционирования системы при наличии механизма предупреждения нарушений

где: n — число моментов контроля среды; λ — интенсивность нарушений характеристик СЗИ.

Определив пределы интегрирования (рис. 3.9) и упростив выражение, получим:

$$F_{\hat{z}}^{s} = \int_{0}^{z} f_{\hat{T}}\left(T\right) e^{-\lambda T}\left(1 - e^{-\lambda T}\right) dT + \int_{z}^{t-z} f_{\hat{T}}\left(T\right) e^{-\lambda T}(1 - e^{-\lambda z}) dT + \\ + \int_{t-z}^{t} f_{\hat{T}}\left(T\right) e^{-\lambda T} dT - e^{-\lambda t}(\frac{z}{t})^{n},$$

Разложив в степенной ряд подынтегральные выражения, получим приближенное значение функции распределения, которое является основным расчетным соотношением:

$$F_{\hat{z}}^{s} \approx n \sum_{i=0}^{r} (\sum_{j=0}^{n-1} b_1 b_2) - e^{-\lambda t}(\frac{z}{t})^{n},$$

где: r — число итераций;

$$b_1 = (-1)^{-I+J} C_{n-1}^{i} \frac{\lambda^{j}}{(j!\, t^{i+1}\left(i + j + 1\right))};$$

$$b_2 = t^{i+j+1} - z^{i+j+1}\ (2^{j} - e^{-\lambda z})(t - z)^{i+j+1}\, e^{-\lambda z}.$$

Сравним полученную стохастическую модель с детерминированной. В детерминированной модели моменты контроля образуют регу-

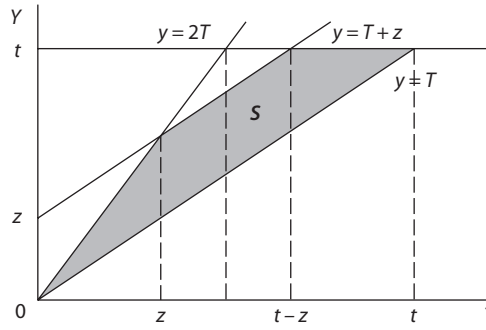

Рис 3.9. Область интегрирования интервала времени предупреждения нарушения

лярный поток с постоянной величиной интервала $T = t/(n+1)$ и следующей плотностью распределения времени предварительного контроля:

$$g_{\hat{z}} = \lambda e^{-\lambda(T+z)}.$$

Следовательно, выражение для функции распределения в детерминированной модели будет иметь следующий вид:

$$F_{\hat{z}}^{d} = e^{-\lambda T}(1 - e^{-\lambda z});\ 0 < z < T.$$

Сравнивая расчетные выражения моделей, при заданных значениях λ, Q, t и n получаем критерий, позволяющий сделать выбор той или иной модели:

$$F_{\hat{Z}}^{S}(z) \lesseqgtr F_{\hat{Z}}^{d}(z).$$

Вероятность P_n предупреждения нарушений за t период жизненного цикла СЗИ можно представить следующим образом:

$$P_n = (n+1) \cdot F_{\hat{Z}},$$

где: n — число моментов контроля; $F_{\hat{z}} = max\left(F_{\hat{z}}^{s}(z), F_{\hat{z}}^{d}(z)\right)$.

Сравнительный анализ стохастических и детерминированных моделей показал эффективность первых при малом числе моментов контроля. Поэтому при управлении информационной безопасностью систем численными методами можно определить предпочтительные модели (стохастическую, детерминированную либо комбинированную), повышающие уровень доверия к СЗИ. Это дает эффект, подобный введению структурной избыточности, то есть можно говорить об особом виде избыточности — стохастической, использование которой практически не увеличивает затрат [52].

4. МЕТОДИКИ СЕРТИФИКАЦИОННЫХ ИСПЫТАНИЙ

Существующие типовые методики сертификационных и аттестационных испытаний носят зачастую описательный характер, что затрудняет автоматизацию и оптимизацию процессов оценки соответствия средств защиты информации [35]. В разделе рассмотрен новый подход к формализации методик сертификационных испытаний СЗИ, позволяющий определить факторы, связанные со временем, стоимостью и полнотой испытаний СЗИ [5,7,8].

4.1. Формальный базис испытаний средств защиты информации

Пусть $R = \{r_i,\}$ – множество требований, предъявляемых к СЗИ Σ, $T = \{t_i\}$ – множество тестовых процедур, проверяющих реализацию предъявляемых требований.

Под *методом разработки тестовых процедур* будем понимать отображение: $M : \Sigma \times R \to T$. Функция M на основе требования $r_i \in R$ и информации о реализации СЗИ Σ. Как правило, функция M для СЗИ Σ является биективным отображением:

– $\forall r_1, r_2 \in R$ выполняется $(r_1 \neq r_2) \Rightarrow (M(\Sigma, r_1) \neq M(\Sigma, r_2))$, то есть различные требования безопасности участвуют в формировании различных тестовых процедур;

– $\forall t \in T \exists r \in R : M(\Sigma, r) = t$ – любая тестовая процедура разрабатывается с целью проверки определенного требования.

Каждая тестовая процедура $t_i \in T$ характеризуется целью выполнения, последовательностью выполняемых действий, результатом выполнения тестовой процедуры и критерием принятия положительного решения.

Цель испытания содержит изложение намерения о выполнении оценки соответствия СЗИ предъявляемым требованиям. *Последова-*

тельность выполняемых действий определяет набор шагов, осуществляемых экспертом для приведения СЗИ в исходное состояние и выполнения тестовой процедуры. *Результаты выполнения тестовых процедур* фиксируются с использованием различных программных средств (ПС) проведения испытаний, таких как: средства генерации и перехвата сетевого трафика, поиска остаточной информации, проверки системы разграничения доступа. *Критерий принятия положительного решения* должен содержать эталонные результаты выполнения тестовых процедур. Введем операторы выполнения требования F_R и корректности выполнения тестовой процедуры F_C.

Оператор выполнения требования r_i для СЗИ $F_R : \Sigma \times R \to \{0,1\}$:

$$F_R(\Sigma, r_i) = \begin{cases} 1, & \text{если требование } r_i \text{ выполнено;} \\ 0, & \text{в противном случае.} \end{cases}$$

Оператор корректности выполнения тестовой процедуры t_i для СЗИ Σ $F_C : \Sigma \times T \to \{0,1\}$:

$$F_C(\Sigma, t_i) = \begin{cases} 1, & \text{если тест } t_i \text{ выполнен успешно;} \\ 0, & \text{в противном случае.} \end{cases}$$

Оператор F_C показывает, что для СЗИ Σ выполнение тестовой процедуры завершилось успешно: фактические результаты, зарегистрированные при выполнении теста, соответствуют эталонным значениям, указанным в описании тестовой процедуры.

Методикой испытаний назовем набор из пяти объектов $\mathbb{A} = \{\Sigma, R, M, F_R, F_C\}$, где R — множество требований, предъявляемых к СЗИ Σ, M — метод разработки тестовых процедур, F_R и F_C — операторы выполнения требования и корректности выполнения тестовой процедуры соответственно, а также для $\forall r_i \in R$ справедливо $F_R(\Sigma, r_i) \Rightarrow F_C(\Sigma, M(\Sigma, r_i))$.

В общем виде испытания включают три стадии: планирование, тестирование, анализ результатов.

При *планировании* выполняется анализ документации и особенностей работы СЗИ. Эксперты должны установить, что в технической документации разработчик декларирует соответствие СЗИ требованиям R, то есть $F_R(\Sigma, r_i) = 1$ для $\forall r_i \in R$. На основании анализа документации, тестовых запусков СЗИ и предъявляемых тре-

бований, формируется множество тестовых процедур $T = \{t_i\}$, где $t_i = M(\Sigma, r_i)$.

Тестирование выполняется с использованием набора тестовых процедур $T = \{t_i\}$, в результате чего для каждой тестовой процедуры определяются результаты выполнение тестовой процедуры, подлежащие регистрации.

На стадии *анализа* фактических и эталонных значений получают множество упорядоченных пар вида ($t_i, F_C(\Sigma, t_i)$). Для СЗИ Σ декларируется соответствие требованиям $R = \{r_i\}$, если:

$$\sum_{i=1}^{n} \left(F_R(\Sigma, r_i) \cdot F_C(\Sigma, M(\Sigma, r_i)) \right) = n,$$

то есть в ходе проведения испытаний установлено соответствие реальных возможностей СЗИ декларируемым в документации или нормативном документе.

4.2. Методика испытаний средств вычислительной техники

Базовым документом, в котором предъявляются требования к комплексу СЗИ, является руководящий документ Гостехкомиссии России по средствам вычислительной техники (см. подр. 4.2.1). Документ устанавливает 7 классов защищенности и содержит требования к реализации дискреционного и мандатного принципов контроля доступа, очистки памяти, изоляции модулей, маркировки документов, защиты ввода и вывода на отчуждаемый физический носитель информации, сопоставления пользователя с устройством, идентификации и аутентификации, регистрации событий, обеспечения целостности и др. Рассмотрим формализованный порядок проверки для указанных наиболее ресурсоемких требований $R_{СЗИ} = \{r_1, r_2, r_3, r_4, r_5, r_6\}$ указанного нормативного документа (см. табл. 4.1.).

4.2.1. Методика проверки дискреционного принципа контроля доступа

Цель выполнения проверки состоит в определении степени соответствия СЗИ требованиям функциональных возможностей по реализации дискреционного принципа разграничения доступа.

Таблица 4.1

Основные требования к средствам вычислительной техники

Обозначение	Требование
r_1	КСЗ должен контролировать доступ наименованных субъектов (пользователей) к наименованным объектам (файлам, программам, томам и т.д.). Для каждой пары (субъект — объект) в СВТ должно быть задано явное и недвусмысленное перечисление допустимых типов доступа (читать, писать и т.д.), т.е. тех типов доступа, которые являются санкционированными для данного субъекта (индивида или группы индивидов) к данному ресурсу СВТ (объекту). КСЗ должен содержать механизм, претворяющий в жизнь дискреционные правила разграничения доступа. Контроль доступа должен быть применим к каждому объекту и каждому субъекту (индивиду или группе равноправных индивидов). КСЗ должен содержать механизм, претворяющий в жизнь дискреционные ПРД, как для явных действий пользователя, так и для скрытых, обеспечивая тем самым защиту объектов от НСД (т.е. от доступа, не допустимого с точки зрения заданного ПРД). Под «явными» здесь подразумеваются действия, осуществляемые с использованием системных средств — системных макрокоманд, инструкций языков высокого уровня и т.д., а под «скрытыми» — иные действия, в том числе с использованием собственных программ работы с устройствами.
r_2	Реализация мандатного принципа контроля доступа. Для реализации этого принципа должны сопоставляться классификационные метки каждого субъекта и каждого объекта, отражающие их место в соответствующей иерархии. Посредством этих меток субъектам и объектам должны назначаться классификационные уровни (уровни уязвимости, категории секретности и т.п.), являющиеся комбинациями иерархических и неиерархических категорий. Данные метки должны служить основой мандатного принципа разграничения доступа. КСЗ должен реализовывать мандатный принцип контроля доступа применительно ко всем объектам при явном и скрытом доступе со стороны любого из субъектов: — субъект может читать объект, только если иерархическая классификация в классификационном уровне субъекта не меньше, чем иерархическая классификация в классификационном уровне объекта, и неиерархические категории в классификационном уровне субъекта включают в себя все иерархические категории в классификационном уровне объекта; — субъект осуществляет запись в объект, только если классификационный уровень субъекта в иерархической классификации не больше, чем классификационный уровень объекта в иерархической классификации, и все иерархические категории в классификационном уровне субъекта включаются в неиерархические категории в классификационном уровне объекта.

r_3	Реализации очистки памяти. КСЗ должен осуществлять очистку оперативной и внешней памяти. Очистка должна производиться путем записи маскирующей информации в память при ее освобождении (перераспределении).
r_4	Изоляция модулей. При наличии в СВТ мультипрограммирования в КСЗ должен существовать программно-технический механизм, изолирующий программные модули одного процесса (одного субъекта), от программных модулей других процессов (других субъектов), т.е. в оперативной памяти программы разных пользователей должны быть защищены друг от друга.
r_6	Защита ввода и вывода на отчуждаемый физический носитель информации. КСЗ должен различать каждое устройство ввода-вывода и каждый канал связи как произвольно используемые или идентифицированные («помеченные»). При вводе с «помеченного» устройства (вывода на «помеченное» устройство) КСЗ должен обеспечивать соответствие между меткой вводимого (выводимого) объекта (классификационным уровнем) и меткой устройства. Такое же соответствие должно обеспечиваться при работе с «помеченным» каналом связи.
r_7	Реализация сопоставления пользователя с устройством. КСЗ должен обеспечивать вывод информации на запрошенное пользователем устройство как для произвольно используемых устройств, так и для идентифицированных (при совпадении маркировки). Идентифицированный КСЗ должен включать в себя механизм, посредством которого санкционированный пользователь надежно сопоставляется выделенному устройству.
r_8	Идентификация и аутентификация. КСЗ должен требовать от пользователей идентифицировать себя при запросах на доступ, должен проверять подлинность идентификатора субъекта — осуществлять аутентификацию. КСЗ должен располагать необходимыми данными для идентификации и аутентификации и препятствовать входу в СВТ неидентифицированного пользователя или пользователя, чья подлинность при аутентификации не подтвердилась.
r_9	Регистрация событий. КСЗ должен быть в состоянии осуществлять регистрацию следующих событий: использование идентификационного и аутентификационного механизма; запрос на доступ к защищаемому ресурсу (открытие файла, запуск программы и т.д.); создание и уничтожение объекта; действия по изменению ПРД. Для каждого из этих событий должна регистрироваться следующая информация: дата и время; субъект, осуществляющий регистрируемое действие; тип события (если регистрируется запрос на доступ, то следует отмечать объект и тип доступа); успешно ли осуществилось событие (обслужен запрос на доступ или нет).
r_{10}	Обеспечение целостности КСЗ. Необходимо осуществлять периодический контроль за целостностью КСЗ.

Введем определения, используемые при описании тестовой процедуры. Пусть $S = \{S_1, S_2, ..., S_i, ..., S_n\}$ – множество тестовых субъектов доступа и $O = \{O_1, O_2, ..., O_j, ..., O_m\}$ – множество тестовых объектов доступа, $R = \{R_1, R_2, ..., R_k, ..., R_l\}$ – множество возможных прав доступа (например, чтение, запись, удаление и т.п.). Определим матрицу доступа $M = (m_{ij})$, $m_{ij} \subseteq R$, где m_{ij} – множество прав доступа тестового субъекта S_i к тестовому объекту O_j. Строка матрицы доступа соответствует субъекту S_i, а столбец – объекту O_j. На пересечении строки и столбца указывается множество прав доступа $m_{ij} \subseteq R$ соответствующего субъекта к данному объекту.

Оператор наличия права доступа субъекта к объекту в матрице $F_M : M, S_i, O_j, R_k \rightarrow \{0,1\}$:

$$F_M(M, S_i, O_j, R_k) = \begin{cases} 1, \text{если у субъекта } S_i \text{ есть право доступа } R_k \text{ к объекту } O_j; \\ 0, \text{в противном случае.} \end{cases}$$

Оператор фактического наличия права доступа субъекта к объекту $F_{fact} : S_i, O_j, R_k \rightarrow \{0,1\}$:

$$F_{fact}(S_i, O_j, R_k) = \begin{cases} 1, \text{если у субъекта } S_i \text{ есть право доступа } R_k \text{ к объекту } O_j; \\ 0, \text{в противном случае.} \end{cases}$$

Последовательность выполняемых действий может быть следующей:

1. Создание тестовых субъектов $S = \{S_1, S_2, ..., S_i, ..., S_n\}$ и объектов доступа $O = \{O_1, O_2, ..., O_j, ..., O_m\}$. Проверка должна проводиться для всех возможных субъектов и объектов доступа, перечень субъектов и объектов определяется на основе анализа документации на СЗИ.

2. Настройка правил разграничения доступа субъектов испытываемого СЗИ к тестовым защищаемым объектам. Данная операция заключается в настройке матрицы доступа $M = (m_{ij})$, $m_{ij} \subseteq R$. В ходе проведения тестирования проверяются все возможные права доступа субъектов к объектам, а также их комбинации.

3. Тестирование настроек СЗИ: проверка фактического наличия права r_k у субъекта S_j по отношению к объекту O_j. Таким же образом определяются все значения оператора $F_{fact}(S_i, O_j, R_k)$ для любых i, j, k. Проверка осуществляется с использованием тестовых попыток доступа (например, чтение, запись, удаление и т. д.) субъектов к объектам.

4. Сравнение фактических прав доступа с требуемыми правами, определенными в матрице доступа.

Результатами выполнения тестовой процедуры, подлежащими регистрации, являются:

1. Множество тестовых субъектов доступа $S = \{S_1, S_2, ..., S_i, ..., S_n\}$, множество тестовых объектов доступа $O = \{O_1, O_2, ..., O_j, ..., O_m\}$, $R = \{R_1, R_2, ..., R_k, ..., R_l\}$ – множество возможных прав доступа.

2. Результаты настройки правил разграничения доступа – матрица доступа $M = (m_{ij})$, $m_{ij} \subseteq R$.

3. Результаты проверки фактического наличия права R_k у субъекта S_j по отношению к объекту O_j – значения оператора $F_{fact}(S_i, O_j, R_k)$ для всех *i, j, k*.

Определим критерий принятия положительного решения: в результате сравнения фактических прав доступа с требуемыми правами, определенными в матрице доступа, должно быть получено совпадение фактических и требуемых прав доступа:

$$F_M(M, S_i, O_j, R_k) = F_{fact}(S_i, O_j, R_k) \text{ для } \forall\ i, j, k.$$

Проверка установленных прав доступа осуществляется путем осуществления попыток «явного» и «скрытого» доступа субъектов к объектам с фиксацией результатов проведенных попыток доступа: успешная или неуспешная.

Анализ полученных данных заключается в сравнении результатов проведенных попыток доступа с ожидаемыми результатами, которые отражены в тестовой матрице доступа.

Аналогично проверке дискреционного разграничения доступа проводится *проверка сопоставления пользователей и устройств*, но объектами доступа в данном случае будут различные устройства ввода-вывода.

4.2.2. Методика проверки мандатного принципа контроля доступа

Введем определения, используемые при описании тестовой процедуры. Пусть $S = \{S_1, S_2, ..., S_i, ..., S_n\}$ – множество тестовых субъектов доступа, $O = \{O_1, O_2, ..., O_j, ..., O_m\}$ – множество тестовых объектов доступа, $M = \{m_1, m_2, ..., m_k, ..., m_l\}$ – множество классификационных меток (классификационных уровней) субъектов и объектов доступа (классификационные метки являются иерархическими: $m_1 < m_2 <, ..., m_{k-1} < m_k < m_{k+1}, ..., m_{l-1} < m_l$), m_{Oi} – классификационная метка *i*-го объекта доступа, m_{Sj} – классификационная метка *j*-го субъекта доступа.

Введем оператор $F_{READ} : S_i, O_j \to \{0,1\}$ проверки наличия права на чтение и оператор $F_{WRITE} : S_i, O_j \to \{0,1\}$ проверки права на запись у субъекта S_i к объекту O_j:

$$F_{READ}(S_i, O_j) = \begin{cases} 1, \text{если у субъекта } S_i \text{ есть право доступа на чтение} \\ \quad \text{к объекту } O_j; \\ 0, \text{в противном случае.} \end{cases}$$

$$F_{WRITE}(S_i, O_j) = \begin{cases} 1, \text{если у субъекта } S_i \text{ есть право доступа на запись} \\ \quad \text{к объекту } O_j; \\ 0, \text{в противном случае.} \end{cases}$$

Последовательность выполняемых действий включает в себя следующие действия:

1. Создание тестовых субъектов $S = \{S_1, S_2, ..., S_i, ..., S_n\}$ и объектов доступа $O = \{O_1, O_2, ..., O_j, ..., O_m\}$.

2. Присвоение классификационных меток $M = \{m_1, m_2, ..., m_k, ..., m_l\}$ субъектам доступа (задается отображение $F_S : S \to M$, которое позволяет вычислять классификационный уровень любого субъекта доступа, т. е. $F_S(S_i) = m_{Si}$).

3. Присвоение классификационных меток $M = \{m_1, m_2, ..., m_k, ..., m_l\}$ объектам доступа (отображение $F_O : O \to M$, позволяет вычислить значения классификационного уровня любого объекта доступа $F_O(O_j) = m_{Oj}$).

4. Выполнение следующих тестовых попыток доступа субъектов к объектам:

– чтение данных из объектов доступа субъектами доступа, т.е. вычисление значений $F_{READ} : S_i, O_j \to \{0,1\}$ для $\forall i, j$;

– запись данных субъектами доступа в объекты доступа, т.е. вычисление значений $F_{WRITE} : S_i, O_j \to \{0,1\}$ для $\forall i, j$.

5. Проверка полученных результатов на соответствие правилам мандатного разграничения доступа.

Результаты выполнение тестовой процедуры, подлежащие регистрации, будут следующими:

1. Множество тестовых субъектов доступа $S = \{S_1, S_2, ..., S_i, ..., S_n\}$, множество тестовых объектов доступа $O = \{O_1, O_2, ..., O_j, ..., O_m\}$, $M = \{m_1, m_2, ..., m_k, ..., m_l\}$ – множество классификационных меток (классификационных уровней) субъектов и объектов доступа.

2. Результаты настройки правил разграничения доступа – отображения $F_S : S \to M$, $F_O : O \to M$ (классификационные метки субъектов и объектов доступа).

3. Результаты проверки фактического наличия прав на запись и чтение у субъекта S_i по отношению к объекту O_j – значения операторов $F_{READ} : S_i, O_j \to \{0,1\}$ и $F_{WRITE} : S_i, O_j \to \{0,1\}$ для $\forall i, j$.

В результате проверки правил мандатного разграничения доступа имеем следующие данные:

– субъект доступа S_j с классификационным уровнем m_{Sj} может осуществлять чтение данных из объекта доступа O_i с классификационным уровнем m_{Oi} тогда и только тогда, когда классификационный уровень субъекта доступа S_j выше, либо равен классификационному уровню объекта доступа O_i, т. е. для $\forall i, j$, $F_{READ}(S_i, O_j) = 1$, тогда и только тогда, когда $m_{Sj} \geq m_{Oi}$;

– субъект доступа S_j с классификационным уровнем m_{Sj} может осуществлять запись данных в объект доступа O_i с классификационным уровнем m_{Oi} тогда и только тогда, когда классификационный уровень объекта доступа O_i выше, либо равен классификационному уровню субъекта доступа S_j, т. е. для $\forall i, j$, $F_{WRITE}(S_i, O_j) = 1$, тогда и только тогда, когда $m_{Oi} \geq m_{Sj}$.

Аналогично проводится *проверка защиты ввода и вывода на отчуждаемый физический носитель информации.*

4.2.3. Методика проверки механизмов очистки памяти

Введем определения, используемые при описании тестовой процедуры. Пусть $A = \{a_1, a_2, ..., a_i, ..., a_l\}$ – множество участков памяти, подлежащих проверке (оперативная память, разделы на жестком диске, внешние носители и т. д.), S – тестовая последовательность символов, используемая при проверке (уникальна для каждого участка памяти А). В ходе описания проверки будем использовать оператор наличия тестовой последовательности в памяти $F_{check} : a_i, S \to \{0,1\}$:

$$F_{check}(a_i, S) = \begin{cases} 1, & \text{последовательность } S \text{ присутствует на участке } a_i; \\ 0, & \text{в противном случае.} \end{cases}$$

Последовательность выполняемых действий следующая:

1. Настройка средств очистки памяти.

2. Помещение тестовых данных в память (уникальная последовательность текстовых символов S).

3. Определение расположения тестовых данных в памяти (адрес, сектор диска и т.д.).

4. Перераспределение (освобождение) памяти с использованием штатных средств испытательного стенда.

5. Контроль наличия тестовых данных в памяти (повторный поиск и обращение по адресам, определенным на начальном этапе) — определение значения $F_{check}(a_i, S)$.

6. Анализ полученных результатов.

7. Применение критериев оценки.

Результаты выполнения тестовой процедуры, подлежащие регистрации, следующие:

1. Результаты настройки средств очистки памяти.

2. Результаты контроля наличия тестовых данных в памяти после ее перераспределения (освобождения).

Критерий принятия положительного решения следующий: последовательность символов *S*, помещенная в память, после освобождения (перераспределения) памяти не была обнаружена, т.е. $F_{check}(a_i, S) = 0, \forall a_i$.

4.2.4. Методика проверки механизмов изоляции модулей

Обеспечение изоляции модулей следует из наличия у каждого процесса (запущенного от имени какого-либо пользователя) отдельного адресного пространства и невозможности прямого доступа к данному пространству процессами, запущенными от имени других пользователей.

Последовательность выполняемых действий:

1. Запуск программ (процессов) от имени различных пользователей.

2. Попытки доступа к памяти процессов, запущенных от своего имени.

3. Попытки доступа к памяти процессов, запущенных от имени других пользователей (субъектов).

4. Попытки доступа к физической памяти ПЭВМ.

5. Анализ результатов.

6. Применение критериев оценки.

Результатами выполнения тестовой процедуры, подлежащими регистрации, являются факты попыток доступа к памяти процессов, запущенных от имени других пользователей, и попыток доступа к физической памяти ЭВМ.

Критерий принятия положительного решения следующий: доступ к памяти процессов, запущенных от имени других пользователей, и к оперативной памяти в ходе испытаний получен не был.

4.2.5. Методика проверки механизмов идентификации и аутентификации субъектов доступа

Введем определения, используемые при описании тестовой процедуры. Пусть A – алфавит паролей и идентификаторов пользователей СЗИ. Обозначим пользователя $id \in ID \subseteq A^*$, пароль – $pwd \in PWD \subseteq A^*$. Учетная запись пользователя $usr_i \in USR$ характеризуется следующим кортежем $usr_i = (id_j, pwd_k)$. Введем оператор корректности учетных данных $F_{AUT} : USR \to \{0,1\}$:

$$F_{AUT}(usr) = \begin{cases} 1, \text{ доступ к СЗИ получен;} \\ 0, \text{ в противном случае.} \end{cases}$$

При проверке корректности реализации механизмов идентификации и аутентификации субъектов может быть использована следующая последовательность выполняемых действий:

1. Включение механизма идентификации и аутентификации СЗИ и создание множества учетных записей субъектов доступа $USR = \{usr_1, usr_2, \ldots\}$.

2. Выполнение запросов на идентификацию и проведение аутентификации с использованием различных сочетаний учетных данных: зарегистрированный/незарегистрированный идентификатор, верный/неверный пароль – $try_i = (id_j, pwd_k)$.

3. Анализ полученных данных.

Результаты выполнение тестовой процедуры, подлежащие регистрации:

1. Конфигурация СЗИ – множество USR учетных записей субъектов доступа.

2. Полученные результаты тестовых запросов на идентификацию и аутентификации: множество $\{F_{AUT}(try_1), F_{AUT}(try_2), \ldots\}$.

Критерии принятия положительного решения:

1. После ввода зарегистрированного идентификатора и пароля пользователю предоставляется доступ к защищаемым ресурсам: $F_{AUT}(try_i) = 1 \Leftrightarrow try_i \in ADM$.

2. После ввода незарегистрированного идентификатора и/или неверного пароля пользователю отказывается в доступе к защищаемым ресурсам: $F_{AUT}(try_i) = 0 \Leftrightarrow try_i \notin ADM$.

Проверка надежности связывания идентификации и аутентификации пользователя со всеми его действиями осуществляется в ходе проведения проверок дискреционного и мандатного принципов разграничения доступа. По окончании каждой из проверок анализируется содержимое журналов регистрации событий.

Проверка считается успешно выполненной, если журнал аудита содержит записи обо всех попытках доступа и всех действиях любых пользователей (в том числе администраторов безопасности), осуществленных в ходе проверок по предыдущим пунктам. При этом в каждой регистрационной записи присутствует информация об идентификаторе пользователя, который был предъявлен при попытке доступа, и/или под которым пользователь был зарегистрирован в системе и выполнял различные действия.

4.2.6. Методика проверки механизмов контроля целостности

Пусть $FILE = \{file_1, file_2, ..., file_n\}$ – множество файлов СЗИ (конфигурационные файлы, программные модули). Введем операторы нарушения целостности F_{MOD} и контроля целостности файлов СЗИ F_{INT}.

Оператор нарушения целостности $F_{MOD} : FILE \to \{0,1\}$:

$$F_{MOD}(file) = \begin{cases} 1, \text{целостность файла нарушается при проведении} \\ \quad \text{испытаний;} \\ 0, \text{в противном случае.} \end{cases}$$

Оператор контроля целостности файлов СЗИ $F_{INT} : FILE \to \{0,1\}$:

$$F_{INT}(file) = \begin{cases} 1, \text{целостность файла нарушена;} \\ 0, \text{в противном случае.} \end{cases}$$

Обозначим $FILE^{\Delta} = \{file_1^{\Delta}, file_2^{\Delta}, ..., file_n^{\Delta}\}$ – множество файлов СЗИ, модифицированных в ходе проведения испытания. При этом выполняется модификация файла $file_i$ в файл $file_i^{\Delta}$. При проверке корректности реализации механизма контроля целостности СЗИ может быть использована следующая последовательность выполняемых действий:

1. Настройка средств контроля целостности (реакция на нарушение целостности, способ контроля целостности, период проверки, условие проверки и т. д.) и идентификация множества файлов СЗИ $FILE = \{file_1, file_2, ...\}$.

2. Внесение изменений в файлы СЗИ (изменение конфигурации, подмена (модификация) исполняемых файлов и т. п.) — получение множества измененных файлов $FILE^{\Delta} = \{file_1^{\Delta}, file_2^{\Delta}, ...\}$.

3. Инициализация проверки целостности файлов СЗИ (создание условий, при которых СЗИ осуществляет контроль целостности).

4. Анализ реакции СЗИ на нарушение целостности своей программной или информационной части.

Результаты выполнения тестовой процедуры, подлежащие регистрации:

1. Множество файлов СЗИ $FILE = \{file_1, file_2, ...\}$.

2. Множество модифицированных файлов СЗИ $FILE^{\Delta} = \{file_1^{\Delta}, file_2^{\Delta}, ...\}$.

3. Реакции СЗИ на нарушение целостности: $F_{INT}(file_1^{\Delta}), F_{INT}(file_2^{\Delta}), ...$

Критерий принятия положительного решения: СЗИ обнаружены все факты нарушения целостности:

$$F_{INT}(file_i^{\Delta}) = F_{MOD}(file_i) \text{ для } \forall i \in [1, n].$$

4.3. Методика испытаний межсетевых экранов

Требования к межсетевым экранам (МЭ) по безопасности информации определены в РД Гостехкомиссии России, в котором указаны пять классов защищенности (см. подр. 2.4.2, табл. 2.5). Каждый класс защищенности МЭ характеризуется минимальной совокупностью требований. Рассмотрим порядок проверки для наиболее ресурсоемких требований $R = \{r_1, r_2, r_3\}$ (см. табл 4.2).

4.3.1. Проверка механизмов фильтрации данных и трансляции адресов

Цель выполнения проверки состоит в определении степени соответствия функциональных возможностей МЭ по фильтрации сетевых пакетов с учетом следующих параметров: сетевого адреса отправителя, сетевого адреса получателя. Исходными данными для формирования тестовой процедуры τ_1 являются множество сетевых адресов, используемых в тестовых сегментах: $IP = IP_S \bigcup IP_R = \{IP_S^1, IP_S^2, \ldots, IP_R^1, IP_R^2\}$. Предполагается, что при проведении тестирования выполняется отправление пакетов из внешнего сегмента сети (сетевые адреса вида IP_S^i) во внутренний сегмент (сетевые адреса вида IP_R^j).

Проверка включает следующие шаги:

Таблица 4.2

Основные требования к межсетевым экранам

Обозначение	Требование
r_1	МЭ должен обеспечивать фильтрацию на сетевом уровне. Решение по фильтрации может приниматься для каждого сетевого пакета независимо на основе, по крайней мере, сетевых адресов отправителя и получателя или на основе других эквивалентных атрибутов.
r_2	МЭ должен обеспечивать идентификацию и аутентификацию администратора МЭ при его запросах на доступ. МЭ должен предоставлять возможность для идентификации и аутентификации по идентификатору (коду) и паролю условно-постоянного действия. Дополнительно МЭ должен препятствовать доступу неидентифицированного субъекта или субъекта, подлинность идентификации которого при аутентификации не подтвердилась.
r_3	МЭ должен содержать средства контроля за целостностью своей программной и информационной части.

1. Настройка правил фильтрации МЭ в соответствии с проверяемым требованием, в результате чего формируется множество запрещающих $RULE^0 = \left\{rule_1^0, rule_2^0, \ldots\right\}$ и разрешающих $RULE^1 = \left\{rule_1^1, rule_2^1, \ldots\right\}$ правил межсетевого экранирования, причем, каждое правило представляет собой упорядоченное множество вида $rule_k^{0/1} = \left(IP_S^i, IP_R^j\right)$, где: IP_S^i – сетевой адрес отправителя, IP_R^j – сетевой адрес получателя.

2. Запуск ПС перехвата и анализа сетевых пакетов во внутреннем и внешнем сегментах сети.

3. Генерация сетевых пакетов из внешнего сегмента сети во внутренний сегмент для всех возможных пар (отправитель, получатель): $packet_k = \left(IP_S^i, IP_R^j, payload^k\right)$.

4. Завершение перехвата сетевых пакетов. В результате получаем следующие множества перехваченных пакетов $PACKET^{IN} = \left\{packet_1^{IN}, packet_2^{IN}, \ldots\right\}$ и $PACKET^{OUT} = \left\{packet_1^{OUT}, packet_2^{OUT}, \ldots\right\}$, где $packet_k^{IN} = \left(IP_S^i, IP_R^j, payload^k\right)$ – сетевые пакеты, перехваченные во

внешнем сегменте, $packet_k^{OUT} = \left(IP_S^i, IP_R^j, payload^k\right)$ – сетевые пакеты, перехваченные во внутреннем сегменте.

5. Экспорт журнала регистрации разрешенных и запрещенных пакетов МЭ. В результате выполнения формируется множество записей о запрещении прохождения $JOUR^0 = \left\{journal_1^0, journal_2^0, \ldots\right\}$ и разрешении прохождения $JOUR^1 = \left\{journal_1^1, journal_2^1, \ldots\right\}$ сетевого пакета. Каждая запись имеет вид: $journal_k^{0/1} = \left(IP_S^i, IP_R^j\right)$.

Результатами выполнения тестовой процедуры являются:

1. Конфигурация МЭ – множества $\left\{rule_i^0\right\}$ и $\left\{rule_i^1\right\}$.

2. Результаты перехвата сетевых пакетов на внешнем и внутреннем интерфейсах МЭ – множества $\left\{packet_i^{IN}\right\}$ и $\left\{packet_i^{OUT}\right\}$.

3. Фрагмент журнала регистрации событий МЭ, демонстрирующий результаты фильтрации сетевых пакетов: множества $\left\{journal_i^0\right\}$ и $\left\{journal_i^1\right\}$.

В качестве критерия принятия положительного решения примем фиксацию факта соответствия реальных (пакеты на входном интерфейсе МЭ, пакеты на выходном интерфейсе МЭ и фрагмент журнала регистрации событий МЭ) и ожидаемых результатов (правила фильтрации МЭ):

$$\begin{cases} PACKET^{OUT} = RULE^1; \\ PACKET^{IN} \;/\; PACKET^{OUT} = RULE^0; \\ PACKET^{OUT} = JOUR^1; \\ PACKET^{IN} \;/\; PACKET^{OUT} = JOUR^0. \end{cases}$$

При проведении тестирования могут быть использованы, например, следующие ПС: *nmap, Packet Generator* (генерация сетевых пакетов), *wireshark, tcpdump* (перехват и анализ сетевых пакетов), программный комплекс *«Сканер-ВС»* (генерация, перехват и анализ сетевых пакетов) [20, 29, 56].

К МЭ предъявляются также аналогичные требования к механизмам фильтрации на других уровнях модели ISO/OSI. Для канального уровня требование выглядит следующим образом: «МЭ должен обеспечивать фильтрацию с учетом входного и выходного сетевого

интерфейса как средство проверки подлинности сетевых адресов». Методика проверки аналогична методике, представленной выше, но в качестве критериев фильтрации используются адреса канального уровня (MAC-адрес). На сетевом уровне проверяется требование по фильтрации с учетом любых значимых полей сетевых пакетов. При проведении испытаний, как правило, внимание следует уделять тестированию механизма фильтрации с учетом следующих полей пакета сетевого уровня: адрес отправителя, адрес получателя, тип протокола верхнего (транспортного) уровня, время жизни пакета (TTL) [27].

4.3.2. Проверка механизмов идентификации и аутентификации администраторов

Целью проверки является определение степени соответствия функциональных возможностей МЭ по идентификации и аутентификации администратора МЭ.

Будем считать, что A – алфавит паролей и идентификаторов администраторов МЭ. Обозначим идентификатор администратора $id \in ID \subseteq A^*$, пароль – $pwd \in PWD \subseteq A^*$. Учетная запись администратора $adm_i \in ADM$ характеризуется следующим кортежем $adm_i = \left(id_j, pwd_k\right)$.

Введем оператор корректности учетных данных $F_{AUT} : ADM \to \left\{0,1\right\}$:

$$F_{AUT}\left(adm\right) = \begin{cases} 1, \text{ доступ на администрирование получен;} \\ 0, \text{ в противном случае.} \end{cases}$$

Предполагается, что идентификация/аутентификация выполняется с использованием сетевых протоколов с ЭВМ внутреннего сегмента сети. Тогда проверка будет включать следующую последовательность действий:

1. Включение механизма идентификации и аутентификации МЭ и создание множества учетных записей администраторов МЭ $ADM = \left\{adm_1, adm_2, \ldots\right\}$.

2. Запуск ПС перехвата и анализа сетевых пакетов во внутреннем сегменте сети.

3. Выполнение запросов на идентификацию и проведение аутентификации с использованием различных сочетаний учетных данных: зарегистрированный/незарегистрированный идентификатор, верный/неверный пароль – $try_i = \left(id_j, pwd_k\right)$.

4. Генерация сетевых пакетов из внутренней сети во внешнюю (или наоборот), прохождение которых разрешается (запрещается) в соответствии с правилами фильтрации МЭ.

5. Завершение перехвата сетевых пакетов, экспорт журнала регистрации событий МЭ.

6. Анализ на предмет наличия учетных данных, передаваемых в открытом виде.

При выполнении проверки реализации локальной идентификации/аутентификации шаги 2, 5 последовательности действий не выполняются.

Результатами выполнения тестовой процедуры являются:

1. Конфигурация МЭ – множество ADM учетных записей администраторов МЭ.

2. Полученные результаты тестовых запросов на идентификацию и аутентификацию: множество $\left\{F_{AUT}\left(try_i\right)\right\}$.

3. Результаты перехвата сетевых пакетов на внешнем и внутреннем интерфейсах МЭ.

4. Фрагмент журнала регистрации событий МЭ, демонстрирующий результаты идентификации и аутентификации.

Определим критерии принятия положительного решения:

1. После ввода зарегистрированного идентификатора и пароля пользователю предоставляется доступ к средствам администрирования МЭ: $F_{AUT}\left(try_i\right) = 1 \Leftrightarrow try_i \in ADM$.

2. После ввода незарегистрированного идентификатора и/или неверного пароля пользователю отказывается в доступе к средствам администрирования МЭ: $F_{AUT}\left(try_i\right) = 0 \Leftrightarrow try_i \notin ADM$.

3. Журнал регистрации событий содержит записи обо всех тестовых попытках получения доступа.

4. Попытки поиска идентификационных данных (имя пользователя, пароль) в перехваченных пакетах не дали результатов.

При проведении тестирования механизмов идентификации/аутентификации для перехвата и анализа сетевых пакетов могут быть использованы, например, программы wireshark, tcpdump, программный комплекс «Сканер-ВС» и др.

4.3.3. Проверка механизмов контроля целостности

Проверка состоит в определении степени соответствия функциональных возможностей МЭ по контролю целостности программной и информационной части МЭ.

Пусть $FILE = \{file_1, file_2, \ldots, file_n\}$ – множество файлов МЭ (конфигурационные файлы, программные модули). Введем операторы нарушения целостности F_{MOD} и контроля целостности файлов МЭ F_{INT}.

Оператор нарушения целостности $F_{MOD} : FILE \to \{0,1\}$:

$$F_{MOD}(file) = \begin{cases} 1, \text{ целостность файла нарушена при проведении испытаний;} \\ 0, \text{ в противном случае.} \end{cases}$$

Оператор контроля целостности файлов МЭ $F_{INT} : FILE \to \{0,1\}$:

$$F_{INT}(file) = \begin{cases} 1, \text{ целостность файла нарушена;} \\ 0, \text{ в противном случае.} \end{cases}$$

Обозначим $FILE^{\Delta} = \{file_1^{\Delta}, file_2^{\Delta}, \ldots, file_n^{\Delta}\}$ – множество файлов МЭ, модифицированных в ходе проведения испытания. При этом выполняется модификация файла $file_i$ в файл $file_i^{\Delta}$. При проверке корректности реализации механизма контроля целостности МЭ может быть использована следующая последовательность выполняемых действий:

1. Включение механизма контроля целостности программной и информационной части МЭ и идентификация множества файлов МЭ $FILE = \{file_1, file_2, \ldots, file_n\}$.

2. Внесение изменений в файлы МЭ (изменение конфигурации, подмена (модификация) исполняемых файлов и т. п.) – получение множества измененных файлов $FILE^{\Delta} = \{file_1^{\Delta}, file_2^{\Delta}, \ldots, file_n^{\Delta}\}$.

3. Инициализация проверки целостности файлов МЭ (создание условий, при которых МЭ осуществляет контроль целостности).

4. Анализ реакции МЭ на нарушение целостности своей программной или информационной части.

Результатами выполнения тестовой процедуры следует считать:

1. Множество файлов МЭ $FILE = \{file_1, file_2, \ldots, file_n\}$.

2. Множество модифицированных файлов МЭ

$$FILE^{\Delta} = \{file_1^{\Delta}, file_2^{\Delta}, \ldots, file_n^{\Delta}\}.$$

3. Реакции МЭ на нарушение целостности:

$$F_{INT}(file_1^{\Delta}), F_{INT}(file_2^{\Delta}), \ldots, F_{INT}(file_n^{\Delta}).$$

В качестве критерия принятия положительного решения примем факт, что обнаружены все факты нарушения целостности:

$$F_{INT}(file_i^{\Delta}) = F_{MOD}(file_i).$$

4.4. Методика испытаний автоматизированных систем

Под автоматизированной системой (АС) будем понимать систему, состоящую из персонала и комплекса средств автоматизации его деятельности, реализующую информационную технологию выполнения установленных функций[38]. Таким образом, АС представляет собой совокупность следующих объектов: средства вычислительной техники, программное обеспечение, каналы связи, информация на различных носителях, персонал и пользователи системы, эксплуатационная и организационно-распорядительная документация.

Руководящий документ Гостехкомиссии России по АС устанавливает классификацию АС, подлежащих защите от НСД к информации, и задаёт требования по защите информации в АС различных классов (см. подр. 2.4.3, табл. 2.6). Рассмотрим формализованный порядок проверки для наиболее ресурсоемких требований $R_{IS} = \{r_1, r_2, r_3\}$ (см. табл. 4.3).

Перед началом проведения тестирования эксперты должны установить, что в технической документации (например, в задании по безопасности на АС) на объект испытаний декларируется соответствие АС требованиям R_{IS}, то есть $F_R(\Sigma, r_i) = 1$ для $\forall r_i \in R_{IS}$.

Таблица 4.3

Основные требования к автоматизированным системам

Обозначение	Требование
r_1	Должна осуществляться идентификация и проверка подлинности субъектов доступа при входе в систему по идентификатору (коду) и паролю условно-постоянного действия, длиной не менее шести буквенно-цифровых символов
r_2	Должен осуществляться контроль доступа субъектов к защищаемым ресурсам в соответствии с матрицей доступа
r_3	Должна быть обеспечена целостность программных средств защиты информации от НСД, а также неизменность программной среды

[38] ГОСТ 34.003-90.

4.4.1. Методика проверки механизмов идентификации и аутентификации субъектов доступа

Проверка выполняется с целью контроля организационных мероприятий, которые позволяют удовлетворить требования к парольной политике, анализу установленных параметров функционирования средств идентификации и аутентификации, контролю корректности функционирования механизмов идентификации и аутентификации, а также контролю процедуры смены паролей пользователями.

Введем определения, используемые при описании тестовой процедуры. Пусть A — алфавит паролей и идентификаторов субъектов доступа (пользователей АС). Обозначим идентификатор пользователя $id \in ID \subseteq A^*$, пароль — $pwd \in PWD \subseteq A^*$. Учетная запись субъекта доступа $s_i \in S$ характеризуется следующим кортежем. Введем оператор корректности учетных данных $F_{AUT} : S \to \{0,1\}$:

$$F_{AUT}(s_i) = \begin{cases} 1, & \text{выполнен вход в систему;} \\ 0, & \text{в противном случае.} \end{cases}$$

При проверке корректности реализации механизмов идентификации и аутентификации может быть использована следующая тестовая процедура.

Тогда проверка будет включать следующую последовательность действий:

1. Проверить наличие эксплуатационных документов на АС, в которых регламентирован порядок проведения парольной защиты АС. Проверить наличие следующих положений:

– требования к сложности паролей (длина, сложность);

– обязанности администратора безопасности по реализации парольной политики АС (генерация паролей, распределение паролей);

– обязанности пользователей по реализации парольной политики АС (генерация паролей, смена паролей).

2. Определить установленные СЗИ от НСД в АС значения для следующих параметров: минимальная длина пароля, сложность пароля (алфавит паролей), максимальный срок действия пароля, максимальное число неудачных попыток входа пользователей в АС, после которого осуществляется блокировка работы пользователя, реакция АС на превышение максимального числа неудачных попыток входа пользователя.

3. Произвольным образом выбрать несколько АРМ и выполнить запросы на идентификацию и проведение аутентификации с исполь-

зованием различных сочетаний учетных данных: зарегистрированный/незарегистрированный идентификатор, верный/неверный пароль – $try_i = \left(id_j, pwd_k\right)$.

4. Под учетными записями пользователей произвести попытки установить пароль, не соответствующий нормативным требованиям. Для этого осуществить:

– попытку установить пароль, длина которого менее 6 символов;

– попытки установить пароль, состоящий исключительно из цифр, либо только из букв.

Результатами выполнения тестовой процедуры, подлежащими регистрации, являются:

1. Положения документации на АС относительно реализации и сопровождения системы парольной защиты.

2. Конфигурация СЗИ от НСД АС в части реализации парольной защиты.

3. Полученные результаты тестовых запросов на идентификацию и аутентификации – множество $\left\{F_{AUT}\left(try_i\right)\right\}$.

4. Полученные результаты тестовых попыток изменения паролей.

В данном случае критериями принятия положительного решения являются следующие:

1. В документах организации, эксплуатирующей АС, установлены требования (сложность, минимальная длина) к паролям пользователей рассматриваемой АС, соответствующие нормативным требованиям.

2. В нормативных документах организации, эксплуатирующей АС, описана процедура действий администратора безопасности по реализации парольной политики АС (процедуры генерация паролей, распределение паролей);

3. Настройки СЗИ от НСД выполнены таким образом, что длина пароля для пользователей АС не может быть менее 6 символов, а установленные ограничения сложности не позволяют использовать пароли, состоящие из однотипных символов.

4. После ввода зарегистрированного идентификатора и пароля пользователю предоставляется доступ к АС: $F_{AUT}\left(try_i\right) = 1 \Leftrightarrow try_i \in S$.

5. После ввода незарегистрированного идентификатора и/или неверного пароля пользователю отказывается в доступе к АС: $F_{AUT}\left(try_i\right) = 0 \Leftrightarrow try_i \notin S$.

6. Все попытки установить пароль, не соответствующий нормативным требованиям, завершились неудачно.

4.4.2. Методика проверки механизмов управления доступом

Целью проверки является определение степени соответствия фактических настроек системы дискреционного разграничения доступа требуемым настройкам, определенным в матрице доступа. Исходными данными для формирования тестовой процедуры являются: множество возможных субъектов доступа $S = \{S_i\}$, множество защищаемых объектов $O = \{O_i\}$, конечное множество прав доступа $P = \{P_i\}$ и матрица доступа.

Последовательность выполняемых действий следующая:

1. Идентификация субъектов (например, пользователей) $S = \{S_i\}$ и объектов доступа (например, объектов файловой системы) $O = \{O_i\}$.

2. Идентификация матрицы доступа субъектов к защищаемым объектам.

3. Для каждой тройки $(S_i, O_j, P_k) \in S \times O \times P$ выполнение тестирования фактического наличия права P_k у субъекта S_i по отношению к объекту O_j (тестирование настроек СЗИ АС).

4. Сравнение фактических прав доступа с требуемыми правами, определенными в матрице доступа.

Результаты выполнения тестовой процедуры, подлежащие регистрации:

1. Матрица доступа субъектов к защищаемым объектам.

2. Фактические результаты тестирования системы дискреционного разграничения доступа.

В качестве критерия принятия положительного решения имеем полученное соответствие фактических и декларируемых прав доступа, определенных в матрице доступа.

4.4.3. Методика проверки механизмов контроля целостности

Целью выполнения процедуры является определение степени соответствия функциональных возможностей СЗИ от НСД АС по контролю целостности программных СЗИ от НСД.

Пусть $FILE = \{file_i\}$ – множество файлов СЗИ от НСД (конфигурационные файлы, программные модули). Введем операторы нарушения целостности – F_{MOD} и контроля целостности файлов СЗИ от НСД – F_{INT}.

Оператор нарушения целостности $F_{MOD} : FILE \to \{0,1\}$:

$$F_{MOD}\left(file\right)=\begin{cases}1, \text{ целостность файла нарушена при проведении испытаний;}\\ 0, \text{ в противном случае.}\end{cases}$$

Оператор контроля целостности файлов СЗИ от НСД $F_{INT}: FILE \to \{0,1\}$:

$$F_{INT}\left(file\right)=\begin{cases}1, \text{ зафиксировано нарушение целостности файла;}\\ 0, \text{ в противном случае.}\end{cases}$$

Обозначим $FILE^{\Delta}=\left\{file_1^{\Delta}, file_2^{\Delta},\ldots, file_n^{\Delta}\right\}$ – множество файлов СЗИ от НСД, модифицированных в ходе проведения испытания. При этом выполняется модификация файла $file_i$ в файл $file_i^{\Delta}$. При проверке корректности реализации механизма контроля целостности может быть использована следующая тестовая процедура.

Последовательность выполняемых действий следующая:

1. Идентификация множества файлов СЗИ от НСД $FILE=\left\{file_1, file_2,\ldots, file_n\right\}$.

2. Внесение изменений в файлы (изменение конфигурации, подмена (модификация) исполняемых файлов и т. п.) – получение множества измененных файлов $FILE^{\Delta}=\left\{file_1^{\Delta}, file_2^{\Delta},\ldots, file_n^{\Delta}\right\}$.

3. Инициализация проверки целостности файлов СЗИ от НСД (создание условий, при которых СЗИ от НСД осуществляет контроль целостности).

4. Анализ реакции СЗИ от НСД на нарушение целостности своей программной или информационной части.

Результатами выполнения тестовой процедуры, подлежащими регистрации, являются:

1. Множество файлов $FILE=\left\{file_1, file_2,\ldots, file_n\right\}$.

2. Множество модифицированных файлов

$$FILE^{\Delta}=\left\{file_1^{\Delta}, file_2^{\Delta},\ldots, file_n^{\Delta}\right\}.$$

3. Реакции СЗИ от НСД на нарушение целостности:

$$F_{INT}\left(file_1^{\Delta}\right), F_{INT}\left(file_2^{\Delta}\right),\ldots, F_{INT}\left(file_n^{\Delta}\right).$$

Критерием принятия положительного решения является факт, что средствами защиты информации обнаружены все факты нарушения целостности:

$$F_{INT}\left(file_i^{\Delta}\right)=F_{MOD}\left(file_i\right) \text{ для } \forall i\in\left[1,n\right].$$

4.5. Методика проведения испытания по требованиям «Общих критериев»

Процедуру проведения оценки в соответствии с ОК в общем виде можно сформулировать следующим образом. Пусть $C = \{c_1, c_2, ..., c_n\}$ – множество компонент требований доверия к безопасности, предъявляемых к ОО Σ. Как правило, множество C формируется с использованием одного из предопределенных ОУД. Для каждой компоненты требования доверия c_i определено множество действий $E^{(i)} = \{e_1^{(i)}, e_2^{(i)}, ..., e_{n_i}^{(i)}\}$ (n_i – число действий оценщика для компоненты c_i), которое должен выполнить оценщик (как правило, аккредитованная испытательная лаборатория) для подтверждения соответствия ОО предъявляемой компоненте c_i. Для каждого действия оценщика $e_j^{(i)}$ разрабатывается множество $S_j^{(i)} = \{s_{j\_1}^{(i)}, s_{j\_2}^{(i)}, ..., s_{j\_m_j^{(i)}}^{(i)}\}$ шагов оценивания – наименьшей структурной единицы работ по оцениванию ($m_j^{(i)}$ – число шагов оценивания для действия оценщика $e_j^{(i)}$).

В табл. 4.4 приведен пример действий оценщика и шагов оценивания для компоненты доверия ATE_IND.2 «Выборочное независимое тестирование».

Следует отметить, что разработка шагов оценивания выполняется экспертами испытательной лаборатории на основе методологии оценки безопасности информационных технологий, представленной в ГОСТ Р ИСО/МЭК 18045–2008 (см.подр.2.7.4).

Сформулируем *метод разработки шагов оценивания*, под которым будем понимать отображение $M : \Sigma \times E \to S$.

Функция M на основе действия оценщика $e_j^{(i)}$ и информации о реализации (свидетельств разработчика) ОО Σ осуществляет генерацию множества шагов оценивания $S_j^{(i)}$, выполняемого для проверки удовлетворения ОО множеству C компонент требований доверия. Как правило, функция M для ОО Σ является биективным отображением.

Оператором корректности выполнения действия оценщика $e_j^{(i)} \in E^{(i)}$ для ОО Σ назовем $F_S : \Sigma \times E \to \{0,1\}$:

$$F_S(\Sigma, e_j^{(i)}) = \begin{cases} 1, \text{ для ОО } \Sigma \text{ все шаги оценивания действия } e_j^{(i)} \\ \quad \text{выполнены успешно;} \\ 0, \text{ в противном случае.} \end{cases}$$

Таблица 4.4

Пример действий оценщика и шагов оценивания

Компонента доверия c_i	**Действие оценщика** $e_k^{(i)}$	**Шаг оценивания** $s_{j\_v}^{(i)}$
ATE_IND.2	*ATE_IND.2.1E* Оценщик должен подтвердить, что представленная информация удовлетворяет всем требованиям к содержанию и представлению свидетельств	*ATE_IND.–1* Оценщик должен исследовать ОО, чтобы сделать заключение, согласуется ли тестируемая конфигурация с оцениваемой конфигурацией, определенной в ЗБ.
		ATE_IND.–2 Оценщик должен исследовать ОО, чтобы сделать заключение, правильно ли он установлен и находится ли в состоянии, которое известно
		ATE_IND.–3 Оценщик должен исследовать набор ресурсов, представленных разработчиком, чтобы сделать заключение, эквивалентны ли они набору ресурсов, использованных разработчиком для функционального тестирования ФБО
	...	...
		...

Процедурой оценки назовем набор из четырех объектов $\mathrm{A} = \left\{\Sigma, C, M, F_S\right\}$, где C – множество компонент требований доверия к безопасности, предъявляемых к ОО Σ, M- метод разработки шагов оценивания, а F_S – оператор корректности выполнения действия оценщика.

Процедура предусматривает наличие трех стадий: планирование, выполнение оценки, а также анализ и оформление результатов оценки (рис. 4.1).

На стадии планирования выполняются задачи получения и анализа исходных данных для проведения оценки. На основании выполненного анализа формируются множества $E^{(i)} = \left\{e_1^{(i)}, e_2^{(i)}, ..., e_{n_i}^{(i)}\right\}$ действий оценщика и соответствующих им шагов оценивания. Выполнение оценки осуществляется с использованием сформированного набора шагов оценивания.

Заключительная стадия предполагает выполнение анализа результатов оценки (сравнение фактических и эталонных результатов). В результате анализа получаем множество упорядоченных пар вида $\left(e_j^{(i)}, F_S(\Sigma, e_j^{(i)})\right)$. Для ОО Σ декларируется соответствие компоненте требования доверия c_i, если в ходе выполнения множества действий оценщика $E^{(i)} = \left\{e_1^{(i)}, e_2^{(i)}, \ldots, e_{n_i}^{(i)}\right\}$ для каждого получены положительные результаты:

$$\sum_{j=1}^{n_i}\left(F_S(\Sigma, e_i^{(j)})\right) = n_i.$$

По результатам проведения оценки оформляется технический отчет об оценке. Для ОО декларируется соответствие требованиям доверия к безопасности $C = \left\{c_1, c_2, \ldots, c_n\right\}$, если $\forall i \in [1, n] \sum_{j=1}^{n_i}\left(F_S(\Sigma, e_i^{(j)})\right) = n_i$.

При проведении оценки, как правило, применяются следующие методы:

– экспертно-документальный;
– функциональное тестирование;
– статический и динамический анализ исходных текстов;
– тестирование проникновением.

Экспертно-документальный метод заключается в проверке соответствия ОО установленным требованиям на основании экспертной

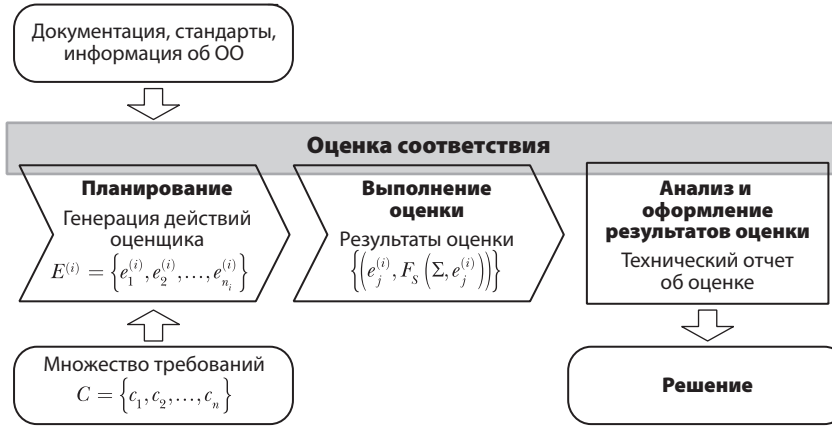

Рис. 4.1. Процедура оценки в соответствии с «Общими критериями»

оценки полноты и достаточности представленных свидетельств и может использоваться для:

– анализа и проверки процессов и процедур, связанных с разработкой и реализацией ОО;

– анализа эксплуатационной документации;

– анализа разработанных функциональных тестов и полученных результатов;

– оценки соответствия параметров ОО исходным требованиям.

Функциональное тестирование заключается в проверке функций безопасности ОО с использованием специализированных тестирующих средств (применяется при независимом функциональном тестировании).

Статический анализ исходных текстов состоит в синтаксическом и семантическом анализе структуры и взаимосвязей функциональных и информационных объектов программ и построении маршрутов выполнения функциональных объектов. Динамический анализ представляет собой динамический контроль выполнения функциональных объектов (ветвей) программы (в отладочном режиме и/или с фиксацией трассы выполнения программы) и используется для детального исследования алгоритма программы. Данные методы могут использоваться для выполнения:

– анализа соответствия между представлениями проекта ОО;

– анализа соответствия каждого представления проекта ОО установленным требованиям;

– анализа программных дефектов;

– проверки корректности представленных доказательств разработчика.

Тестирование проникновением заключается в санкционированной попытке обойти существующий комплекс средств защиты ОО. В ходе тестирования оценщик играет роль злоумышленника, мотивированного на нарушение информационной безопасности.

4.6. Рекомендации по оптимизации испытаний

Основной проблемой оценки соответствия СЗИ является экспоненциальный рост временных и материальных затрат на выполнение типовых операций при увеличении сложности разрабатываемых СЗИ и при большом количестве платформ и сред, на которых данные СЗИ могут функционировать. Например, при проведении проверки механизма дискреционного разграничения доступа необходимо выполнить $|S| \cdot |O| \cdot |R|$ типовых операций (см. разд. 4.3). Можно показать,

что время τ_Σ, затрачиваемое на проведение испытаний, носит экспоненциальный характер роста: $\tau_\Sigma \sim nv^w$, где n — число проверяемых требований, w — число факторов тестирования (например, «учетная запись пользователя»), v — число возможных значений фактора тестирования.

В общем случае задача оптимизации процедуры оценки соответствия может быть сформулирована следующим образом.

Пусть $\tau : E \times \Sigma \to N_0$ — это время, затрачиваемое оценщиками на выполнение проверки ОО Σ с использованием действия оценщика $e_j^{(i)}$. Будем считать, что отображение вида $G : C \times \Sigma \to N_0$ представляет затраты на проведение оценки ОО Σ требованиям C.

Решение задачи оптимизации может быть определено как получение минимума времени тестирования при ограничениях на затраты:

$$\begin{cases} \sum_i \sum_j \tau\left(e_j^{(i)}, \Sigma\right) \to \min; \\ \sum_i G\left(c_i, \Sigma\right) \le G_M, \end{cases}$$

где: G_M — ограничения, накладываемые на затраты.

Опыт проведения сертификационных испытаний СЗИ позволил определить ряд методических рекомендаций, позволяющих существенно снизить затраты на оценку соответствия, например:

- совмещение проверок,
- использование выборочного контроля и анализа риска [59],
- использование комбинаторного покрытия ,
- использование средств автоматизации.

При проведении независимого функционально тестирования рекомендуется выполнять совмещение некоторых видов испытаний. Например, процедура тестирования подсистемы регистрации событий может быть совмещена с процедурой тестирования подсистемы разграничения доступа и идентификации/аутентификации.

Существенное сокращение работ может быть достигнуто при обосновании возможности проведения выборочного контроля, который позволяет применение процедуры проверки менее чем к 100% совокупности контролируемых элементов (например: свидетельств разработчика, тестируемых функций безопасности). При использовании методов выборочного контроля необходимо установить приоритеты проверяемых требований с целью дальнейшего определения необходимого количества тестируемых объектов.

Для сокращения временных затрат и повышения качества отчетных материалов при проведении испытаний необходимо использовать программные средства, позволяющие автоматизировать процедуры оценки соответствия. Для независимого тестирования можно использовать как инструментальные средства, широко представленные на современном рынке программного обеспечения, так и программы собственной разработки, написанные на языках сценариев (например, perl или python), для анализа уязвимостей – сканеры безопасности, для документирования – программы типа Doxygen, для проведения анализа исходных текстов – анализаторы безопасности кода [2, 4, 19, 56, 71, 79].

4.7. Рекомендации по контролю отсутствия недекларированных возможностей

Контроль отсутствия недекларированных возможностей ПО СЗИ является наиболее проблемным, трудоемким и ответственным видом испытаний [13, 41, 54, 59, 62, 66, 68, 79]. Требования к сертификационным испытаниям ПО на отсутствие НДВ определены в РД «Защита от несанкционированного доступа к информации Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей» (Гостехкомиссия России, 1999), который рассмотрен в подр. 2.4.4. Для того чтобы выполнить все требования, указанные в таблице 2.7, рассмотрим порядок проведения испытаний на отсутствие НДВ для 2-го уровня контроля.

4.7.1. Общий порядок проведения испытаний

Общий порядок проведения испытаний по контролю отсутствия НДВ следующий:

1. Контроль состава и содержания документации;
2. Контроль исходного состояния ПО (идентификация объекта испытаний);
3. Статический анализ исходных текстов;
4. Динамический анализ исходных текстов;
5. Формирование отчетных документов.

В случае доработки ПО, при выявлении НДВ все этапы должны повторяться для всех изменившихся модулей исходных текстов программ.

В состав документации, представляемой заявителем, в соответствии с требованиями РД должны входить:

— Спецификация (ГОСТ 19.202–78), содержащая сведения о составе ПО и документации;

— Описание программы (ГОСТ 19.402–78), содержащее основные сведения о составе (с указанием контрольных сумм файлов, входящих в состав ПО), логической структуре и среде функционирования ПО, а также описание методов, приемов и правил эксплуатации средств технологического оснащения при создании ПО;

— Описание применения (ГОСТ 19.502–78), содержащее сведения о назначении ПО, области применения, применяемых методах, классе решаемых задач, ограничениях при применении, минимальной конфигурации технических средств, среде функционирования и порядке работы;

— Пояснительная записка (ГОСТ 19.404–79), содержащая основные сведения о назначении компонентов, входящих в состав ПО, параметрах обрабатываемых наборов данных (подсхемах баз данных), формируемых кодах возврата, описание используемых переменных, алгоритмов функционирования;

— Исходные тексты программ (ГОСТ 19.401–78), входящих в состав ПО.

Контроль состава документации должен проводиться экспертами ИЛ путем сравнения перечня представленных документов с требованиями руководящего документа.

Контроль содержания документации осуществляется, как по соответствию формальным требованиям ГОСТ к содержанию составных частей документов, так и по соответствию реальным возможностям ПО.

Контроль исходного состояния ПО заключается в фиксации исходного состояния ПО и сравнении полученных результатов с результатами, приведенными в документации на продукт, например, в Формуляре или Технических условиях.

Результатами контроля исходного состояния должны быть рассчитанные уникальные значения контрольных сумм модулей дистрибутива и исходных текстов программ, входящих в состав сертифицируемого программного изделия. Контрольные суммы должны рассчитываться для каждого файла, входящего в состав сертифицируемого продукта.

Общий порядок проведения сертификационных испытаний на отсутствие НДВ приведен на рис. 4.2.

Порядок проведения контроля полноты и отсутствия избыточности исходных текстов ПО приведен на рис. 4.3.

Идентичность файлов загрузочного кода, собранных из одних и тех же исходных текстов программ, может быть проконтролирована абсолютно или фактически. Абсолютная идентичность свидетельствует об их равенстве от первого до последнего бита. Доказательством абсолютной идентичности двух файлов (исполняемых мо-

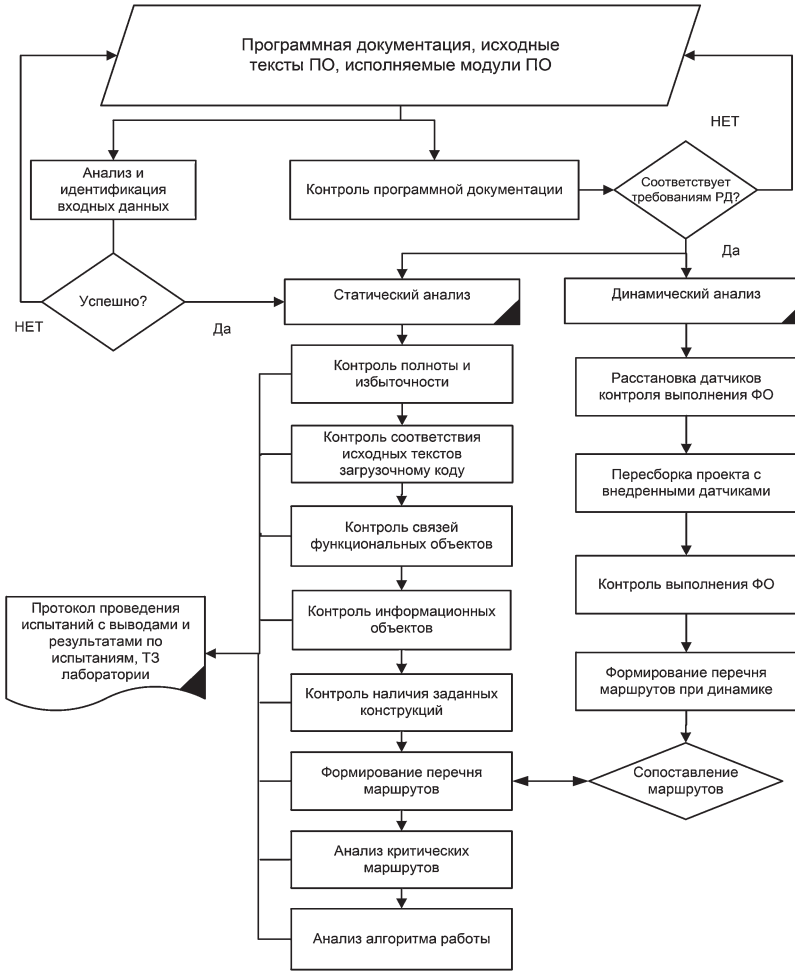

Рис. 4.2. Схема проведения сертификационных испытаний

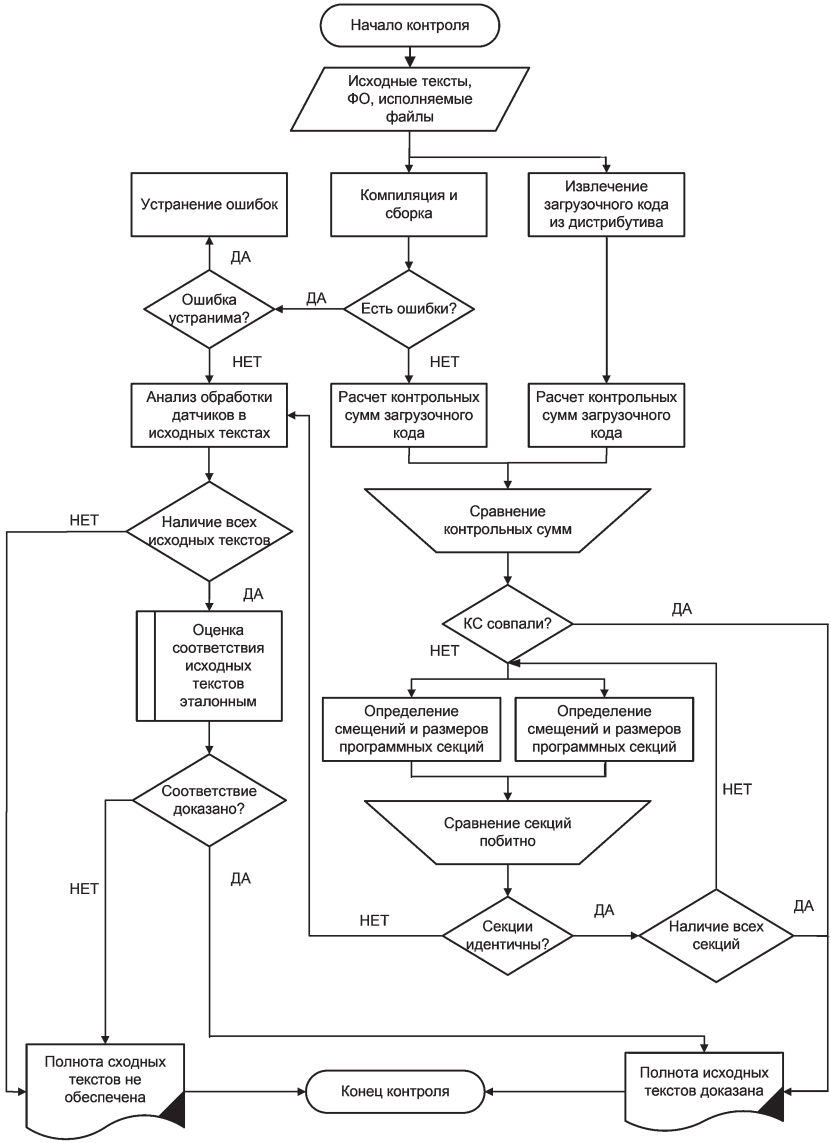

Рис. 4.3. Схема анализа полноты и отсутствия избыточности исходных текстов программ

дулей), кроме побитового сравнения, может быть равенство их контрольных сумм, рассчитанных по сертифицированным или иным доверенным алгоритмам. Фактическая идентичность свидетельствует об идентичности тех секций файлов, которые реализуют функционал программы, то есть секций кода и данных. Доказательством фактической идентичности двух файлов (исполняемых модулей) может быть абсолютная идентичность их секций кода и данных. Абсолютная идентичность указанных секций может быть доказана как сравнением их контрольных сумм, рассчитанных по сертифицированным или иным доверенным алгоритмам, так и побитовым сравнением. Успешное выполнение проверки по указанному методу подтверждает целесообразность дальнейших испытаний исходного текста ПО.

Идея данного контроля связей функциональных объектов по управлению и по информации заключается в построении связей функциональных объектов (модулей, процедур, функций) и получении информации о модульной структуре программного средства и о логической структуре отдельного программного модуля. Для изображения модульной структуры используются следующие характеристики:

— граф вызовов;
— список путей вызовов;
— матрица вызовов и достижимости;
— точки вызовов.

Для изображения логической структуры можно использовать характеристики управляющих графов и трассы выполнения функциональных объектов. В целях обеспечения достаточности приведенных характеристик, модульную структуру дополняют метрикой, позволяющей ранжировать вызовы функций и процедур; к логической структуре добавляют метрику, измеряющую сложность управляющей структуры; а к текстовым характеристикам — метрику уровня реализации. Критериями оценки наличия НДВ в ПО при использовании контроля связей функциональных объектов по управлению могут являться:

— отсутствие недостижимых и незавершенных маршрутов;
— высокая оценка уровня реализации в вершинах управляющего графа;
— отсутствие соответствия формальных решений с признаками дефектов.

Таким образом, данный метод дает дополнительную возможность экспертной оценки структуры исходных текстов ПО.

При контроле информационных объектов различных типов производится построение списка информационных объектов (локаль-

ных переменных, глобальных переменных, внешних переменных) с указанием их принадлежности к функциональным объектам. Контроль информационных объектов различных типов считается успешно выполненным, если в результате анализа списка информационных объектов недекларированных возможностей не выявлено, а построенные списки соответствуют алгоритму работы ПО, приведенного в документации.

Контроль наличия заданных конструкций в исходных текстах заключается в синтаксическом контроле наличия заданных конструкций в исходных текстах ПО из базы потенциально опасных программных конструкций (сигнатур) с использованием специальных средств автоматизации [55]. К указанным конструкциям кода относят фрагменты, содержащие операции, непосредственно связанные с изменением атрибутов безопасности, и системные операции, потенциально способные повлиять на безопасность системы [6, 13, 30, 31, 32, 54, 62, 100].

База сигнатур для поиска программных закладок и уязвимостей должна включать конструкции, например, способные привести к переполнению буфера, реализации «логических бомб», реализации троянских атак, компрометации аутентификационных данных, возможному превышению полномочий, реализации DOS-атак, нелегитимной низкоуровневой работе с дисками и файловой системой, реализации скрытых каналов передачи информации и др. (см. подр. 4.7.2).

При выявлении потенциально опасных конструкций (сигнатур) в исходных текстах необходимо исследовать факт возможности эксплуатации данных конструкций на предмет нарушения доступности, целостности и конфиденциальности в сертифицируемом продукте [59].

На рисунке 4.4 приведена структурная схема метода автоматизации, позволяющего проводить контроль наличия заданных конструкций в исходных текстах.

При формировании перечня маршрутов ПО должны быть построены маршруты выполнения функциональных объектов, процедур, функций, ветвей. Визуальное представление данных маршрутов может быть оформлено в виде графов, матриц, списков или таблиц. Формирование перечня маршрутов позволяет экспертам испытательной лаборатории проследить реальную последовательность выполнения процедур, функций или ветвей и сравнить с тем, как данная информация описана в документации на сертифицируемое изделие.

После формирования перечня маршрутов проводится анализ критических маршрутов выполнения функциональных объектов.

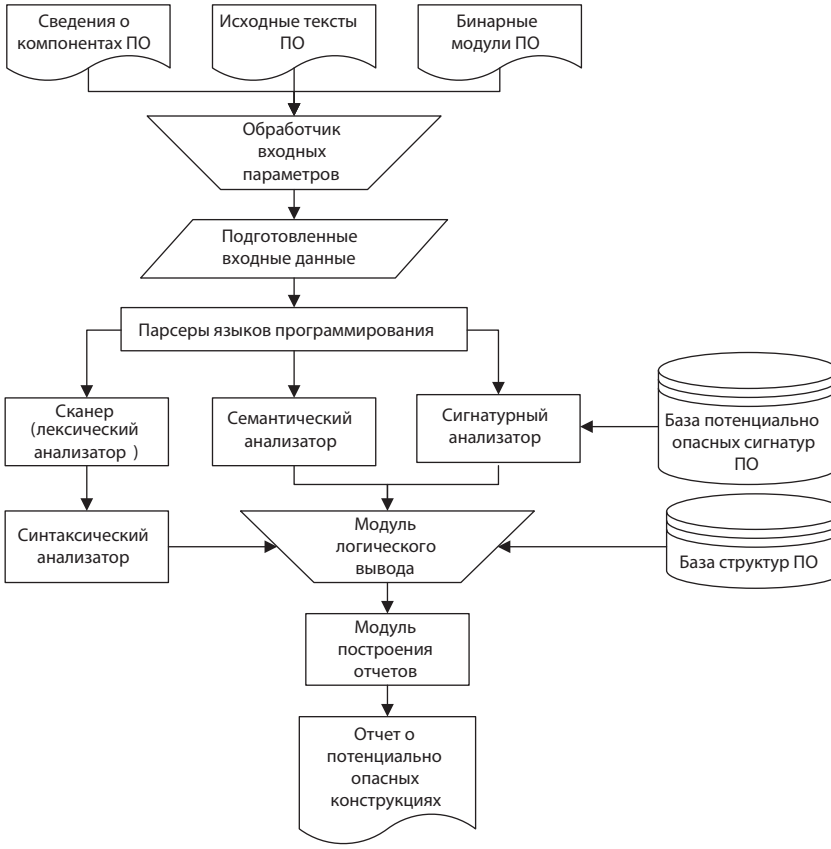

Рис. 4.4. Схема проведения контроля наличия заданных конструкций в исходных текстах

В качестве критических маршрутов выполнения должны рассматриваться маршруты выполнения функциональных объектов, в которых происходит обработка защищаемой информации. Защищаемая информация может содержать критически важные сведения о ПО (пароли, лицензии, работа с базой данных). По результатам анализа должно быть установлено соответствие порядка выполнения критических маршрутов заявленному алгоритму функционирования программного продукта, представленного в документации. Также к критическим маршрутам должны быть отнесены те маршруты, в которых выполняются функциональные объекты, попавшие в список потенциально опасных конструкций.

Анализ алгоритма работы функциональных объектов на основе блок-схем (диаграмм), построенных по исходным текстам контролируемого ПО, заключается в построении по исходным текстам ядра контролируемого ПО и модулей, отвечающих за безопасность, блок-схем с использованием средств автоматизации. В дальнейшем должен быть проведен сравнительный анализ алгоритма работы функциональных объектов и алгоритма работы, приведенного в программной документации на изделие.

Порядок проведения динамического анализа исходных текстов проиллюстрирован на рис. 4.3.

Методика проведения динамического анализа исходных текстов программ включает следующие технологические операции:

— контроль выполнения функциональных объектов (процедур, функций, ветвей);

— сопоставление фактических маршрутов выполнения функциональных объектов (процедур, функций, ветвей) и маршрутов, построенных в процессе проведения статического анализа.

4.7.2. Рекомендации по контролю наличия заданных конструкций

Наиболее важным пунктом РД по НДВ можно назвать требование по контролю наличия заданных конструкций в исходных текстах программ, так как оно касается непосредственно безопасности ПО. Конечной целью этой проверки является выявление критических уязвимостей, особенно преднамеренного характера (программных закладок). Для 2-го уровня контроля отсутствия НДВ необходимо выполнять синтаксический, а для 1-го уровня — семантический контроль наличия заданных конструкций в исходных текстах ПО из базы данных потенциально опасных конструкций (ПОК). Указанные ПОК, по сути, являются признаками возможного присутствия в ПО дефектов безопасности. Как правило, такими признаками могут быть:

— обращения к запрещенным объектам, например: вызовы устаревших функций, использование нестойких криптографических примитивов, обращения к API, не выполняющего определенные требования безопасности);

— типичные ошибки программистов, проявляющиеся на уровне синтаксиса программного кода, внутренней структуры программной системы или внешних связей приложения и позволяющие, в конце концов, снизить уровень целостности, доступности и конфиденциальности информации.

Для выявления дефектов, идентифицируемых как преднамеренные, в базу ПОК добавляют шаблоны, связанные с:

— программным кодом проверки условий активации программных закладок, например: вызовами функций считывания даты и времени, созданием счетчиков обращений к ресурсам, обращениям к настройкам системы;

— программным кодом реализации программных закладок, касающихся разного рода деструктивных функций;

— элементами механизмов безопасности;

— элементами реализации скрытого управления и передачи информации;

— «необычными» объектами программ;

— «необычными» функциями;

— наличием кода, интенсивно использующего ресурсы системы.

Надо понимать, значительная часть дефектов безопасности специфична для каждой системы программирования, поэтому необходимо поддерживать базу данных ПОК для всех видов используемых платформ разработки, причем в актуальном состоянии.

В настоящее время отсутствуют нормативные требования к содержимому базы ПОК. Однако в мировой практике существуют ряд таксономий дефектов и уязвимостей ПО, а именно [58]:

1. CWE — Common Weaknesses Enumeration («Общий реестр недостатков» компании MITRE);

2. Fortify Seven Pernicious Kingdoms (7 разрушительных «царств» компании HP Fortify);

3. OWASP Top Ten (10 самых крупных угроз Open Web Application Security Project — «Открытого проекта безопасности веб-приложений»).

Текущая версия классификации CWE 2.1 включает в себя 617 типов дефектов безопасности ПО, разделенных на 104 категории и отображенных на 22 иерархических вида для разного типа пользователя (Java-программист, аудитор безопасности, веб-программист и т.п.).

Fortify Seven Pernicious Kingdoms представляет собой классификацию дефектов ПО, ориентированную на языки программирования: ABAP, ColdFusion, COBOL, C/C++, C#/VB.NET/ASP.NET, HTML, Java/JSP, Javascript, PHP, Python, PLSQL/TSQL, Visual Basic/VBScript/ASP, XML. К каждому из этих языков программирования поставлен в соответствии специальный раздел, содержащий дефекты из 7 групп («царств» в терминологии HP Fortify): валидации ввода и представления, эксплуатации API, механизмов безопасности, времени и состоя-

ния, обработки ошибок, качества кода, инкапсуляции, а также окружения.

В рамках проекта OWASP разработаны различные артефакты, документы и программные средства для анализа защищенности. Из ключевых документов, созданных в рамках проекта OWASP, следует назвать: AppSec FAQ (ответы на распространенные вопросы по веб-безопасности), Guide to Building Secure Web Applications (руководство по главным аспектам безопасности веб-приложений), OWASP Application Security Verification Standard 2009 (стандарт проверки веб-приложений), Top Ten Web Application Security Vulnerabilities (десятка наиболее актуальных уязвимостей веб-приложений за последний год).

Общий процесс поиска наличия заданных конструкций включает 4 этапа, итеративных между собой:

— выбор новых объектов для исследований;
— запуск средств автоматизация анализа кода;
— рецензирование кода экспертом;
— изучение результатов.

Ключевыми факторами успеха в аудите крупных программных систем являются разделение программного проекта на подсистемы и компоненты с определением их интерфейсов и выделение приоритетов анализа (расположение кода, тип дефектов и т.п.). Определенным подспорьем здесь могут послужить различные метрики, полученные из анализа ПО (плотность дефектов, SLOC, число внешних вызовов и т.п.).

В случае использования средств автоматизации выявления ПОК условно разделяют синтаксический и семантический контроль заданных конструкций.

В случае синтаксического контроля поиск осуществляется либо непосредственно в исходных текстах, либо в наборе распознанных лексем языка программирования. В данном случае сигнатуры могут представлять собой перечень наборов символов или лексем. Этот подход обеспечивает быстрое и легкое создание сигнатур потенциально опасных конструкций и их анализаторов, что удобно при поиске некорректностей кодирования, но мало эффективно при поиске сложных программных закладок. Семантический анализ является по сути «надстройкой» над синтаксическим. После того, как были определены лексемы в исходных текстах программного проекта, на их основе строится, как правило, абстрактное синтаксическое дерево кода. Анализ узлов этого дерева позволяет определять высокоуровневые понятия, например: тип используемой переменной, наличие обращения

к базе данных, направление передачи информации между объектами. В литературе все виды анализа, обрабатывающие результаты синтаксического анализа исходных текстов в каком-либо контексте, относят к семантическому анализу [23].

Чтобы лучше иллюстрировать подход, используемый в семантическом анализе, приведем таблицу по эвристикам обнаружения некоторых ПОК из состава таксономии CWE [71] (табл.4.5):

Таблица 4.5

Примеры эвристик выявления потенциально опасных конструкций

Элемент CWE верхнего уровня	Элемент CWE нижнего уровня	Признаки наличия	Методика обнаружения
Проблемы поведения (Behavioral Problems) (438)	Изменение поведения в новой версии окружения (Behavioral Change in New Version or Environment) (438)	*<совпадение вызова со списком проблемных для зависимости, используемой в проекте>*	Определить версию среды функционирования; сравнить вызов со списком функций, изменивших свое поведение для заданной версии среды
	Ошибки бизнес-логики (Business Logic Errors) (840)	Нет	Требуется построение бизнес-модели кода и её сравнение с бизнес-процессом (плохо поддается автоматизации, требуется рецензирование кода экспертом)
	Нарушение ожидаемого поведения (Expected Behavior Violation) (440)	Нет	Требуется построение бизнес-модели кода и её сравнение с бизнес-процессом (плохо поддается автоматизации, требуется рецензирование кода экспертом)
	Неверный контроль частоты взаимодействия (Improper Control of Interaction Frequency) (799)	*<нет таймаута или запрета попыток в интерактивной функции>* (взаимодействующей с пользователем или внешней стороной)	Определить интерактивность функционального объекта (по наличию API заданного типа); определить внутри интерактивного объекта наличие API по обработке задержки
Ошибки каналов и путей (Channel and path errors) (417)	Недостаточная защита выбранного пути (Improper Protection of Alternate Path) (424)	*<отсутствие проверки при показе закрытой страницы>*	Определить вызов функционала отображающего содержимое страницы, удостовериться в наличии функционала контроля сессии доступа пользователя

Продолжение табл. 4.5

Элемент CWE верхнего уровня	**Элемент CWE нижнего уровня**	**Признаки наличия**	**Методика обнаружения**
Ошибки каналов и путей (Channel and path errors) (417)	Скрытый канал памяти (Covert storage channel) (515)	*<обращение к нестандартным информационным объектам>* (файл подкачки, реестр, прямой доступ к диску, операции с «кучей»)	Поиск вызовов функционала из списка объектов потенциально скрытых каналов
	Скрытый канал времени (Covert timing channel) (385)	*<обращение к нестандартным информационным объектам>* (бесконечные циклы, работа с очередью сообщений, таймером, высокопроизводительным таймером)	Поиск вызовов функционала из списка объектов потенциально скрытых каналов
Обработка данных (Data Handling) (19)	Обработка данных (Data Handling) (19)	*<поиск включения введенных пользователем данных при формировании строк>*	Поиск вызовов получения пользовательских данных; поиск вызовов потенциально опасных функций с использованием пользовательских данных
Обработка ошибок (Error Handling) (388)	Обработка ошибок (Error Handling) (388)	*<наличие в блоке catch операции вывода в журнал>*	Поиск обработчиков исключений и определение внутри них вызовов вывода в журнал
Ошибки обработчика (Handler Errors) (429)	Ошибки обработчика (Handler Errors) (429)	*<выброс исключения, у которого нет обработчика>*	Создание списка всех выбросов исключений; создание списка всех обработчиков; сравнение двух списков
Злоупотребление API (API Abuse) (227)	Злоупотребление API (API Abuse) (227)	*<использование внешних компонент (библиотек) с некорректными версиями>*	Составление списка всех внешних зависимостей с версиями; поиск потенциально опасных вызовов из этих библиотек
Индикатор плохого качества кода (Indicator of Poor Code Quality) (398)	Присваивание вместо (Assigning instead of Comparing) (481)	*<наличие «=» вместо «==» в блоке if, где сравниваются лишь переменные>*	Проверка заданного списка регулярных выражений для содержимого заданных типов блоков

Продолжение табл. 4.5

Элемент CWE верхнего уровня	Элемент CWE нижнего уровня	Признаки наличия	Методика обнаружения
Ошибки очистки и инициализации (Initialization and Cleanup Errors) (452)	Неверная инициализация (Improper Initialization) (665)	*<отсутствие инициализации значений объектов перед их использованием>*	Поиск объявлений объектов, которые требуют инициализации перед своим использованием; поиск первого использования объекта в вызовах и операциях; при отсутствии их инициализации (левая часть в присваивании, функции, использующие их в виде выходных значений)
Недостаточная инкапсуляция (Insufficient Encapsulation) (485)	Остаточный отладочный код (Leftover Debug Code) (489)	*<наличие отладочного кода>*	Поиск присутствия вызовов функций из списка отладочных, наличия подозрительных правил в названиях отладочных функций, ключевых слов по debug, в т.ч. в комментариях
Проблемы указателей (Pointer Issues) (465)	Разыменование нулевых указателей (NULL Pointer Dereference) (476)	*<вызов операций по освобождению памяти для указателей, которые равны NULL>*	Проверка вызовов функций, возвращающих указатель (на которую может повлиять пользователь), и факта проверки значения следом за этим
Механизмы безопасности (Security Features) (254)	Использование жестко прописанных паролей (Use of hard-coded password) (259)	*<наличие паролей и других данных авторизации непосредственно в коде приложения>*	Проверка вызовов функций с параметрами авторизации, использующих аргументы со строковыми константами; поиск функций, требующих авторизации вместе с переменными, которым были присвоены строковые константы
Время и состояние (Time and State) (361)	Внешнее влияние на сферу определения (External Influence of Sphere Definition) (673)	*<проверка наличия функций, использующих значения переменных окружения>*	Поиск вызовов функций, читающих содержимое переменных окружения>
Ошибки интерфейса пользователя (User Interface Errors) (445)	Нереализованная или неподдерживаемая функция в интерфейсе пользователя (Unimplemented of unsupported feature in UI) (447)	*<наличие скрытых элементов GUI, а также некорректностей в обработчиках событий>*	Поиск наличия disabled=true hidden=true и других подобных свойств в файле GUI, контроль отсутствия кода в функции-обработчике (Qt, Builder)

ЛИТЕРАТУРА

1. *Аграновский А.В., Мамай В.И., Назаров И.Г., Язов Ю.К.* Основы технологии проектирования систем защиты информации в информационно-телекоммуникационных системах. Ростов н/Д: Изд-во СКНЦ ВШ, 2006. 258 с.

2. *Адамова Л.Е., Амарян М.Р., Варламов О.О., Лысаковский В.А.* Об одном подходе к созданию ревизоров обеспечения безопасности информации на отдельных компьютерах // Известия Таганрогского государственного радиотехнического университета. 2003. № 4 (33). С. 175–176.

3. *Александрович А.Е., Бородакий Ю.В., Чуканов В.О.* Проектирование высоконадёжных информационно-вычислительных систем. М.: Радио и связь, 2004. 144 с.

4. *Баландин А.В., Ващенко А.П., Войнов Ю.В., Мукминов В.А.* Об архитектуре средств тестирования автоматизированных систем в условиях моделирования информационных воздействий // Известия Института инженерной физики. 2011. № 1 (19). С. 36–39.

5. *Барабанов А.В., Гришин М.И., Марков А.С.* Формальный базис и метабазис оценки соответствия средств защиты информации объектов информатизации. // Известия института инженерной физики. 2011. № 3. С. 82–88.

6. *Барабанов А.В., Марков А.С., Фадин А.А.* Сертификация программ без исходных текстов // Открытые системы. СУБД. 2011. № 4. С.38–41.

7. *Барабанов А.В., Марков А.С., Цирлов В.Л.* Методический аппарат оценки соответствия автоматизированных систем требованиям безопасности информации // Спецтехника и связь. 2011. № 3. С.48–53.

8. *Барабанов А.В., Марков А.С., Цирлов В.Л.* Разработка методики испытаний межсетевых экранов по требованиям безопасности информации. // Вопросы защиты информации. 2011. № 3. С.19–24.

9. *Барабанов А.В., Марков А.С., Цирлов В.Л.* Сертификация систем обнаружения вторжений // Открытые системы. СУБД. 2012. № 3. С.31–33.

10. *Баранов А.П., Зегжда Д.П., Зегжда П.Д., Ивашко А.М., Корт С.С., Кузьмич В.М., Медведовский И.Д., Семьянов П.В.* Теория и практика обеспечения информационной безопасности. М.:: Яхтсмен, 1996. 300 с.

11. *Барсуков В.С., Дворянкин С.В., Шеремет И.А.* Безопасность связи в каналах телекоммуникаций. // Технологии электронных коммуникаций. 1992. Том 20. 122 с.

12. *Бахтизин В.В., Глухова Л.А.* Стандартизация и сертификация программного обеспечения. Мн.: БГУИР, 2006. 200 с.

13. *Беляков И.А., Еремеев М.А.* Подход к построению подсистемы принятия решения о наличии/отсутствии недекларированных возможностей в сертифицируемом

программном обеспечении на основе данных статического анализа // Проблемы информационной безопасности. Компьютерные системы. 2008. № 4. С. 66–72.

14. *Благодаренко А.В., Лисова Ю.В., Тумоян Е.П.* Исследование безопасности гетерогенных корпоративных сетей на основе методологии OSSTMM // Информационное противодействие угрозам терроризма. 2010. № 14. С. 132–135.

15. *Бородакий Ю.В., Добродеев А.Ю., Пальчун Б.П., Лоскутов А.А.* Страхование и обеспечение информационной безопасности // Информационное противодействие угрозам терроризма. 2006. № 6. С. 59–64.

16. *Бородакий Ю.В., Жуков И.Ю., Зубарев И.В., Костогрызов А.И., Родионов В.Н. и др.* Методическое руководство по оценке качества функционирования информационных систем. М.: Изд-во 3 ЦНИИ МО РФ, 2003. 352 с.

17. *Бородакий Ю.В., Тарасов А.А.* О функциональной устойчивости информационно-вычислительных систем // Известия Южного федерального университета. Технические науки. 2006. № 7 (62). С. 5–12.

18. *Варенница В.В.* Проблемы вычисления метрик сложности программного обеспечения при проведении аудита безопасности кода методом ручного рецензирования // Вестник МГТУ им.Н.Э.Баумана. Сер. «Приборостроение». 2011. Спецвыпуск «Технические средства и системы защиты информации». С.79–84.

19. *Васильева М.А., Марков А.С.* Утилиты скрытого наблюдения: классификация, методы работы и выявление // РКТ: Фундаментальные и прикладные проблемы. Секция 7. Информационные технологии и безопасность в ракетно-космических системах. Изд-во МГТУ им.Н.Э.Баумана. 2005. С.100–101.

20. *Ващенко А.П., Кондаков С.Е., Хабибуллин И.В., Фадин А.А.* Мобильное место администратора информационной безопасности // Безопасные информационные технологии. Сборник трудов I НТК. М.: МГТУ им.Н.Э.Баумана. 2010. С.28–35.

21. *Волканов Д.Ю., Зорин Д.А.* Исследование применимости моделей оценки надёжности для разработки программного обеспечения с открытым исходным кодом // Прикладная информатика. 2011. № 2. С.26–32.

22. *Герасименко В.А.* Защита информации в автоматизированных системах обработки данных. // В 2-х кн. М.: Энергоатомиздат, 1994. 175 с.; 400 с.

23. *Глухих М.И., Ицыксон В.М.* Программная инженерия. Обеспечение качества программных средств методами статического анализа. СПб.: Изд-во Политехн. ун-та, 2011. 150 с.

24. *Гончаров И.В., Костина Л.В., Платонов М.С.* Построение обобщенной математической модели типового объекта информатизации на основе использования мультиномиального распределения // Информация и безопасность. 2010. № 4 (13). С. 611–614.

25. *Горшков Ю.Г., Бельфер Р.А.* Анализ риска информационной безопасности сетей при выполнении функций защиты приватности // Электросвязь. 2012. № 3. С.26–28.

26. *Грибков В.В., Федорец О.Н.* Модели и методы разработки безопасного программного обеспечения. М.: Изд-во 3 ЦНИИ МО РФ, 2009. 188 с.

27. *Гудков О.В., Марков А.С.* Методика анализа безопасности межсетевого экрана // РКТ: фундаментальные и прикладные проблемы. Секция 7. Информационные технологии и безопасность в ракетно-космических системах. Изд-во МГТУ им.Н.Э.Баумана. 2005. С. 67–69.

28. *Дидюк Ю.Е., Лютиков В.С., Макаров О.Ю., Рогозин Е.А.* К вопросу оценки уязвимости информации в целях задания требований к средствам защиты информации в автоматизированных системах // Телекоммуникации. 2003. № 4. С. 42–44.

29. *Дорофеев А.В.* Тестирование на проникновение: демонстрация одной уязвимости или объективная оценка защищенности? // Защита информации. Инсайд. 2010. № 6. С.72–74.

30. *Егоров М.А., Марков А.С.* Анализ безопасности исходного кода ПО с использованием упорядоченных бинарных диаграмм решений // Безопасные информационные технологии. Сборник трудов II НТК. М.: МГТУ им.Н.Э.Баумана. 2011. С. 33–36.

31. *Жидков И.В., Львов В.М.* Повышение гарантии выявления программных закладок в программном обеспечении автоматизированных систем военного назначения // Известия Таганрогского государственного радиотехнического университета. 2003. № 4 (33). С. 53–57.

32. *Жилкин С.Д.* Использование моделей поведения для выявления НДВ // Известия Института инженерной физики. 2011. № 1 (19). С. 2–7.

33. *Зима В.М., Котухов М.М., Ломако А.Г., Марков А.С., Молдовян А.А.* Разработка систем информационно-компьютерной безопасности. СПб: ВАК им. А.Ф. Можайского, 2003. 327 с.

34. *Зубарев И.В.* Сертификация как направление повышения безопасности информационных систем и программного обеспечения. // Известия Таганрогского государственного радиотехнического университета, 2003. № 4 (33). С. 48–53.

35. *Кадушкин И.В.* Предложения по совершенствованию методического аппарата проведения испытаний средств защиты информации автоматизированных систем. // Информационное противодействие угрозам терроризма. 2008. № 11. С.195–203.

36. *Карповский Е.Я., Чижов С.А.* Надежность программной продукции. Киев: Тэхника, 1990. 159 с.

37. *Керножицкий В.А., Марков А.С.* Ускоренный метод оценки параметров технических систем по малым статистическим выборкам // Вопросы повышения качества управления движущимися объектами. СПб: БГТУ им. Д.Ф. Устинова, 1995. С. 71–78.

38. *Климов С.М., Михайлов Р.А., Пальчун Б.П.* Техническое регулирование в области обеспечения информационной безопасности // Известия Южного федерального университета. Технические науки. 2003. № 4 (33). С. 40–44.

39. *Клянчин А.И.* Семантический анализ исходного кода для построения модели прикладной области на практике. // Вестник МГТУ им.Н.Э.Баумана. Сер. «Приборостроение». 2011. Спецвыпуск «Технические средства и системы защиты информации». С.85–89.

40. *Ковалев В.В., Компаниец Р.И., Маньков Е.В.* Экспертиза и защита кода программ на основе автоматов динамического контроля // Защита информации. Инсайд. 2007. № 3. С. 48–55.

41. *Колесников Д.В., Петров А.Ю., Храмов В.Ю.* Методика оценки защищенности специального программного обеспечения при проведении испытаний автоматизированных систем // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. 2010. № 1. С.74–79.

42. *Колозезный Э.А., Бужинский В.А., Динеев В.Г., Ковригин М.И., Колозезный А.Э., Можаров Г.П.* Независимая экспертиза – основа сертификации программно-математического обеспечения изделий ракетно-космической техники // Космонавтика и ракетостроение. 2001. Вып. 24. С. 154–162.

43. *Королев В.Ю.* Прикладные задачи теории вероятностей: модели роста надежности модифицируемых систем. М.: Диалог-МГУ, 1997. 68 с.

44. *Костогрызов А.И., Липаев В.В.* Сертификация функционирования автоматизированных информационных систем. М.: Изд. «Вооружение. Политика. Конверсия», 1996. 280 с.

45. *Котенко И.В., Котухов М.М., Марков А.С.* и др. Законодательно-правовое и организационно-техническое обеспечение информационной безопасности автоматизированных систем и информационно-вычислительных сетей. СПб.: ВУС, 2000. 190 с.

46. *Котухов М.М., Кубанков А.Н., Калашников А.О.* Информационная безопасность: Учебное пособие. М.: Академия ИБС–МФТИ, 2009. 195 с.

47. *Марков А.С.* Анализ атак на сети Novell Netware, основанный на таксономии угроз информационной безопасности // Известия вузов. Приборостроение. 2000. Т.43. № 4. С.30–34.

48. *Марков А.С.* Исследование дефектов операционной системы Windows // Известия вузов. Приборостроение. 2000. Т.43. № 6. С.46–50.

49. *Марков А.С.* Модели оценки и планирования испытаний программных средств по требованиям безопасности информации. Вестник МГТУ им.Н.Э.Баумана. Сер. «Приборостроение». 2011. Спецвыпуск «Технические средства и системы защиты информации». С.90–103.

50. *Марков А.С.* Нечеткая модель оценки надежности и безопасности функционирования программного обеспечения по результатам испытаний. // Вестник МГТУ им.Н.Э.Баумана. Сер. «Приборостроение». 2011. Спецвыпуск «Технические средства и системы защиты информации». С.146–151.

51. *Марков А.С.* Оценка динамической сложности программного обеспечения на ПЭВМ // Методы и средства совершенствования сложных управляющих систем и комплексов: Учебное пособие. СПб: Мех. ин-т им. Д.Ф. Устинова (Военмех), 1992. С. 35–47.

52. *Марков А.С.* Парадигма ограниченного стохастического контроля // Известия института инженерной физики. 2012. № 1 (23). С. 15–19.

53. *Марков А.С., Годердзишвили Г.М.* Модель оценки степени отлаженности программного обеспечения по результатам испытаний // Методы и средства управления и контроля. Л.: МО СССР, 1987. С. 51–52.

54. *Марков А.С., Миронов С.В., Цирлов В.Л.* Выявление уязвимостей в программном коде // Открытые системы. СУБД. 2005. № 12. С.64–69.

55. *Марков А.С., Миронов С.В., Цирлов В.Л.* Выявление уязвимостей программного обеспечения в процессе сертификации // Известия Таганрогского государственного радиотехнического университета. 2006. № 7 (62). С. 82–87.

56. *Марков А.С., Миронов С.В., Цирлов В.Л.* Опыт тестирования сетевых сканеров уязвимостей // Информационное противодействие угрозам терроризма. 2005. № 5. С.109–122.

57. *Марков А.С., Никулин М.Ю., Цирлов В.Л.* Сертификация средств защиты персональных данных: революция или эволюция? // Защита информации. Инсайд. 2008. № 5. С.20–25.

58. *Марков А.С., Фадин А.А., Цирлов В.Л.* Систематика дефектов и уязвимостей программного обеспечения // Безопасные информационные технологии. Сборник трудов II НТК. М.: МГТУ им.Н.Э.Баумана. 2011. С. 83–87.

59. *Марков А.С., Цирлов В.Л.* Аудит программного кода по требованиям безопасности. Часть 2.// Information Security. 2008. № 3. С.46—47.

60. *Марков А.С., Цирлов В.Л.* Сертификация программ: мифы и реальность// Открытые системы. СУБД. 2011. № 6. С.26—29.

61. *Марков А.С., Цирлов В.Л.* Управление рисками — нормативный вакуум информационной безопасности// Открытые системы. СУБД. 2007. № 8. С. 63—67.

62. *Марков А.С., Щербина С.А.* Испытания и контроль программных ресурсов// Information Security. 2003. № 6. С.26.

63. *Матвеев В.А., Медведев Н.В., Троицкий И.И., Цирлов В.Л.* Состояние и перспективы развития индустрии информационной безопасности Российской Федерации в 2011 г.// Вестник МГТУ им.Н.Э.Баумана. Сер. «Приборостроение». 2011. Спецвыпуск «Технические средства и системы защиты информации». С.—6.

64. Математические модели и алгоритмы процессов эксплуатации сложных систем / Под общ. ред. В.А. Чобаняна и В.В.Линника. М.: ВА РВСН им. Петра Великого, 2005. 168 с.

65. *Машкина И.В., Рахимов Е.А.* Модель объекта информатизации// Информационное противодействие угрозам терроризма. 2006. № 6. С. 82—87.

66. *Минаков В.А., Мирошников В.В., Ламинский Е.Ю.* Применение аппарата сетей Петри при контроле программного обеспечения на отсутствие недекларированных возможностей// Телекоммуникации. 2009, № 10. С.38—41.

67. *Минзов А.С., Кольер С.М.* Обоснование управленческих решений в сфере обеспечения информационной безопасности// Системный анализ в науке и образовании. 2010. № 1. С. 48—53.

68. *Николаенко Ю.Н., Шубинский И.Б.* Развитие технологии сертификационных испытаний программного обеспечения по требованиям безопасности информации// Надежность. 2010. № 1. С. 66—79.

69. Нормативные и методические документы по технической защите информации. Специальные нормативные документы: официальный сайт ФСТЭК России.— URL: http://www.fstec.ru/_razd/_karto.htm. Дата обращения: 01.09.2012.

70. *Пальчун Б.П., Юсупов Р.М.* Оценка надежности программного обеспечения. СПб.: Наука, 1994. 84 с.

71. Патент 114799 Российская Федерация. Система для определения программных закладок. / В.В. Вылегжанин, А.Л. Маркин, А.С. Марков, Р.А. Уточка, А.А. Фадин, А.К. Фамбулов, В.Л. Цирлов — № 2011153967/08 (081180); заявл. 29.12.11; опубл. 10.04.12, Бюл. № 10. 2 с.

72. *Половко А.М., Гуров С.В.* Основы теории надежности. СПб.: БХВ-Петербург, 2006. 702 с.

73. *Рыжиков Ю.И.* Эффективность и эксплуатация программного обеспечения ЭЦВМ. Л.: МО СССР, 1985. 263 с.

74. *Рыжков Е.А., Карпов А.Н.* Подходы к верификации и тестированию 64-битных приложений// Информационные технологии. 2008. № 7. С.41—45.

75. Сводный отчёт по безопасности программного обеспечения в России и мире. М.: НПО «Эшелон», 2012. Вып.2.— URL: http://cnpo.ru/report_echelon_2012.pdf. Дата обращения: 01.09.2012.

76. *Смагин В.А.* Основы теории надежности программного обеспечения. СПб.: ВКА им.А.Ф.Можайского, 2009. 336 с.

77. *Солодовников В.В., Тумаркин В.И.* Теория сложности и проектирование систем управления. М.: Наука, 1990. 168 с.

78. *Тейер Т., Липов М., Нельсон Э.* Надежность программного обеспечения. М.: Мир, 1981. 325 с.

79. *Темнов О.Д.* Анализ и исследование методов и средств обнаружения недекларированных возможностей // Научно-технический вестник Санкт-Петербургского государственного университета информационных технологий, механики и оптики. 2007. № 39. С. 45–50.

80. *Трубачев А.П., Долинин М.Ю., Кобзарь М.Т., Сидак А.А., Сороковников В.И.* Оценка безопасности информационных технологий / Под ред. В.А. Галатенко, Предисловие С.И. Григорова, М.СИП РИА, 2001. 356 с.

81. *Ухлинов Л.М., Сычев М.П., Скиба В.Ю., Казарин О.В.* Обеспечение безопасности информации в центрах управления полетами космических аппаратов. М.: МГТУ им.Н.Э.Баумана, 2000. 366 с.

82. *Холдстед М.* Начала науки о программах. М.: Финансы и статистика, 1981. 128 с.

83. *Хорев А.А.* Аттестация объектов информатизации и выделенных помещений // Специальная техника. 2006. № 4. С. 49–61.

84. *Цирлов В.Л.* Основы информационной безопасности. Краткий курс. Ростов н/Д, Изд-во: Феникс, 2008. 254 с.

85. *Черкесов Г.Н.* Надежность аппаратно-программных комплексов. СПб.: Питер, 2005. 479 с.

86. *Шаракшанэ А.С., Халецкий А.К., Морозов И.А.* Оценка характеристик сложных автоматизированных систем. М.: Машиностроение, 1993. 271с.

87. *Шахалов И.Ю.* Лицензия как продукт осознанной необходимости. // Защита информации. Инсайд. 2010. № 2. С.53–55.

88. *Штрик А.А., Осовецкий Л.Г., Мессих И.Г.* Структурное проектирование надежных программ встроенных ЭВМ. Л.: Машиностроение, 1989. 296 с.

89. *Alguliev R.M., Imamverdiev Y.N.* About one method of risk measurement of maintenance of information security of corporative networks because of fuzzy sets. // Proc. of the 4th International Conference on New Information (Technologies»2000). Minsk. 2000. V.1. P. 76–81.

90. *Bubnov V., Tyrva A., Khomonenko A.* Model of reliability of the software with Coxian distribution of length of intervals between the moments of detection of errors // Proceedings of 34th Annual IEEE Computer Software and Applications Conference COMPSAC-2010. 2010. P. 238–243.

91. *Gatsenko O. Yu.* Method for the design of information protection systems // Automatic Control and Computer Sciences. 2000. T. 34. № 5. P. 1–8.

92. *Grusho A., Knyazev A., Timonina E.* Strictly consistent tests for detection of statistical covert channels // Journal of Mathematical Sciences. 2007. T. 146. № 4. P. 5984–5991.

93. *Kostogryzov A., Nistratov G., Kleshchev N.* Mathematical Models and Software Tools to Support an Assessment of Standard System Processes. // Proceedings of the 6th International SPICE Conference on Process Assessment and Improvement (SPICE-2006). Luxembourg. 2006. P. 63–68.

94. *Kotenko I., Stepashkin M.* Analyzing Vulnerabilities and Measuring Security Level at Design and Exploitation Stages of Computer Network Life Cycle.// Lecture Notes in Computer Science. 2005. V.3685. P. 317–330.

95. *Lipaev V.V.* Problems of the development and quality control of large software systems // Programming and computer software. 2005. V. 31. № 1. P. 47–49.

96. *Lovtsov D.A.* Data Protection Methods for Automatic Controls for Complicated Dynamic Systems // Automatic Documentation and Mathematical Linguistics. 2000. Vol. 34. № 3. P. 18–37.

97. *Markov A.S., Kernozitsky V.A.* Economically effective data bases diagnostics method // Advances in Modeling and Analysis (AMSE Press). 1995. B: Signals, Information, Data, Patterns. Vol.33. № 3. P. 5–11.

98. *Pozin B., Giniyatullin R., Galakhov I., Vostrikov D.* Model-based Technology of Automated Performance Testing // SYRCoSE. 2009. P. 93.

99. *Utkin L.V., Zatenko S.I., Coolen F.P.A.* New interval Bayesian models for software reliability based on non-homogeneous Poisson processes //Automation and Remote Control. 2010. V. 71. № 5. P. 935–944.

100. *Zegzhda P.D., Zegzhda D.P., Kalinin M.O.* Vulnerabilities detection in the configurations of MS Windows operating system // Lecture Notes in Computer Science. 2005. V. 3685 LNCS. P. 339–351.

А.С. Марков, В.Л. Цирлов, А.В. Барабанов

Методы оценки несоответствия средств защиты информации

Издательство «Радио и связь»
www.radiosv.ru

Подписано в печать 24.06.2012. Формат 60×90/16. Бумага офсет. Печ. л. 12.
Тираж 1000 экз. Заказ №
Отпечатано в