УДК: 004.056
DOI 10.53403/9785951505200_20

ПОВЫШЕНИЕ УРОВНЯ ЗАЩИЩЕННОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ЗА СЧЕТ ВНЕДРЕНИЯ ПРОЦЕССОВ БЕЗОПАСНОЙ РАЗРАБОТКИ

Сергеева Оксана Александровна (*il@vniief.ru*), *Алексеева Екатерина Сергеевна*

ФГУП «РФЯЦ-ВНИИЭФ», г. Саров Нижегородской обл.

В статье рассмотрены способы внедрения безопасной разработки программного обеспечения на стадиях жизненного цикла. Проанализированы основные положения ГОСТ Р 56939 «Защита информации. Разработка безопасного программного обеспечения. Общие требования». Описаны практические аспекты внедрения требований стандарта [1].

Ключевые слова: безопасная разработка; разработка безопасного ПО; повышение уровня защищенности; устранение уязвимостей.

INCREASING SOFTWARE SECURITY LEVEL BY IMPLEMENTING SECURE DEVELOPMENT PROCESSES

Sergeeva Oksana Alexandrovna (*il@vniief.ru*), *Alekseeva Ekaterina Sergeevna*

FSUE «RFNC-VNIIEF», Sarov Nizhny Novgorod region

The article considers the ways of implementation of the secure software development at lifecycle stages. The main provisions of the requirement standard (GOST) 56939-2016 «Information protection. Secure software development. General requirements» were analyzed. The practical aspects of standard requirements implementation were described.

Key words: secure development, development of secure software, increasing the security level, removing vulnerabilities.

Введение

Современный этап мирового развития характеризуется непрерывной цифровизацией и повсеместной автоматизацией аспектов деятельности человека. Сложно представить работу какой-либо сферы деятельности без применения информационных технологий (ИТ). Практически любая задача, встающая перед нами в повседневной жизни, может быть полностью или частично решена посредством того или иного программного обеспечения (ПО). Рынок изобилует различными решениями, от примитивных утилит до мощных систем искусственного интеллекта. Но наряду с развитием ИТ наблюдается высокий рост киберпреступности, атак на защищенные сети и объекты с целью хищения, повреждения или модификации ценных ресурсов.

Атаки могут быть направлены на любую составляющую информационной системы (ИС): сервера, рабочие станции, ноутбуки, мобильные устройства, маршрутизаторы, каналы связи. Нарушитель находится в постоянном поиске, поочередно проверяя каждый отдельный компонент, ведь какой бы неприступной не казалась система защиты, ее стойкость определяется по слабейшему звену.

Как правило, система защиты ИС состоит из комбинаций организационных, технических и программных мер. Огромный пласт атак направлен на эксплуатацию уязвимостей ПО, будь то системные, прикладные или специализированные средства. Под уязвимостью [5] будем понимать недостаток (слабость) программного (программно-технического) средства или ИС в целом, который (которая) может быть использована для реализации угроз безопасности информации. Проблема уязвимости ПО – одна из первостепенных задач в области информационной безопасности, т. к. является основной причиной успешности компьютерных атак, а также причиной разного рода непреднамеренных отказов и потери ресурсов [13]. В российской нормативной базе уяз-

вимости ПО подразделяют на программные закладки [6], скрытые каналы [7], бреши/уязвимости [8] и недекларированные возможности (РД Гостехкомиссии России).

С каждым годом увеличивается количество прикладного и специализированного ПО и, соответственно, растет количество уязвимостей ПО и их разновидностей. На сегодняшний день, по данным компании Hewlett Packard можно выделить более 500 классов различных уязвимостей ПО [12, 14]. При классификации уязвимостей используются такие признаки как область происхождения уязвимости, типы недостатков ИС, место возникновения уязвимости [11]. К наиболее опасным уязвимостям можно отнести уязвимости кода (языков программирования), архитектуры и конфигурации.

Классификация уязвимостей ПО ведется мировым компетентным сообществом. Следует отметить такие крупнейшие базы уязвимостей как Common Vulnerabilities and Exposures (CVE), National Vulnerability Database (NVD), Open Sourced Vulnerability Database (OSVDB), Common Attack Pattern Enumeration and Classification (CAPEC), Common Weakness Enumeration (CWE) [18, 19]. В России учет и классификация уязвимостей осуществляется на государственном уровне.

С 2014 г. ФСТЭК России в лице ГНИИИ ПТЗИ поддерживает собственный реестр известных угроз информационной безопасности и уязвимостей ПО – Банк данных угроз (БДУ) безопасности информации ФСТЭК России. БДУ аккумулирует информацию об уязвимостях: текстовое описание уязвимости, дату обнаружения уязвимости, название, версию и производителя уязвимого ПО, информацию о типе ошибки, классе уязвимости и ее текущем статусе. Также записи содержат:

– оценку критичности уязвимости и сопутствующий вектор CVSS (Common Vulnerability Scoring System);

– пометку о наличии известных сценариев эксплуатации уязвимости и возможного результата ее эксплуатации;

– указание уязвимых аппаратных платформ или операционных систем;

– список возможных методов противодействия уязвимости и ссылки на дополнительные источники информации, включая идентификаторы данной уязвимости в иных реестрах и базах данных.

В продолжение темы классификации стоит отметить альтернативный подход к структурированию информации об обнаруженных уязвимостях ПО – регистрация не самих уязвимостей, а сценариев их эксплуатации (эксплойтов, exploits) или примеров эксплуатации уязвимости (Proof of Concept). Примером реализации такого подхода является база Exploit Database (https://www.exploit-db.com/), сформированная организацией Offensive Security Team, специализирующейся на проведении тренингов в области информационной безопасности и тестирования компьютерных систем на проникновение [19, 20].

Методы и меры противодействия атакам

Совершенствуются не только всевозможные способы несанкционированного воздействия злоумышленниками, но и методы защиты от подобного рода воздействия. Меры противодействия уязвимостям ПО при его эксплуатации можно разделить на два способа.

Первый – применение наложенных средств защиты информации. Это внедрение организационных мер и использование различных программно-технических средств защиты таких как антивирусные программы, автоматизированные средства анализа защищенности, межсетевые экраны, системы глубокого анализа сетевого трафика и другие. Такой подход к решению проблемы не всегда является качественным, но практически всегда дорогостоящим, т. к. доверенные средства защиты должны пройти сертификацию по требованиям безопасности информации, что, как правило, влечет за собой экспоненциальный рост стоимости таких продуктов.

Второй способ нацелен на повышение уровня защищенности самого эксплуатируемого ПО. Такой подход подразумевает доработку самого ПО, т. е. в расширении основного функционала механизмами защиты. Зачастую это емкая задача, требующая больших временных и трудовых ресурсов, т. к. ряд уязвимостей может быть устранен лишь глобальной перестройкой продукта практически «с нуля». Процесс доработки существующего ПО несет ряд отрицательных последствий. «Накрученная» на существующие решения безопасность, хоть и повышает уровень защищенности ПО, не всегда является эффективной. Механизмы защиты работают не всегда корректно, отсутствует «хорошая» документация.

По данным исследований некоторых компаний, в том числе компании IBM, стоимость устранения уязвимости или ошибки в ПО зависит от стадии жизненного цикла ПО, на котором их обнаружили. Каждая из компаний действовала независимо. Однако, результаты получены практически одинаковые: если стоимость усилий, необходимых для обнаружения и устранения уязвимости ПО на стадии дизайна и архитектуры принять за единицу, то те же действия на стадии технической поддержки возрастут в цене в несколько десятков раз: см. рис. 1 [15, 16].

	Дизайн и архитектура	Разработка	Тестирование	Инсталляция	Техническая поддержка
Стоимость, X - приведенная единица стоимости, может выражаться в человеко-часах, денежных единицах и т.д.	**1X**	**5X**	**10X**	**15X**	**30X**

Рис. 1. Стоимость устранения ошибок ПО на стадиях жизненного цикла

Доработка ПО на стадии поддержки и обслуживания несет большие потери финансовых, трудовых и временных ресурсов, что отражено на рис. 2 [20].

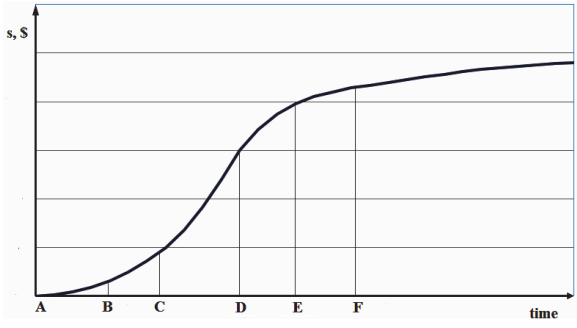

Рис. 2. График затрат на устранение уязвимостей

На графике можно отметить несколько значимых точек: A: уязвимость ПО появилась на стадии постановки требований; B: уязвимость найдена менее чем за один день после первой стадии; C: уязвимость найдена в процессе разработки ПО; D: уязвимость найдена в релизе; E: уязвимость найдена в существующем релизе и влияет на другие части кода ПО; F: уязвимость существует уже давно, и разработчик больше не работает в компании.

Линия отображает сумму всех затрат на исправление уязвимости с течением времени. Наиболее известны потери компании Toyota, обнаруженные в продукте после релиза. В 2009 г. компании Toyota отзывала проданные автомобили в связи с заклиниваем педали акселератора. Компания Toyota обновила ПО автомобилей, но к 2010 г. количество отозванных автомобилей подходило к отметке 8,2 млн. Не трудно представить масштаб затрат на исправление этой ошибки [20].

Исходя из практических данных, можно уверенно сказать, что каждый из обозначенных подходов не дает стопроцентного результата и не способен полностью исключить возможность реализации ошибки или уязвимости в ПО. Существует еще один вариант развития событий – это внедрение методов и мер по безопасной разработке последовательно на каждой фазе жизненного цикла ПО, начиная с проектирования. Данный способ апробирован ведущими мировыми компаниями-разработчиками ПО, которые активно развиваются и укрепляют свои позиции на мировом рынке, предоставляя пользователям защищенные и доверенные продукты.

Осваивание данного метода в России только начинает набирать обороты, и регуляторы в области защиты информации полностью поддерживают и даже настоятельно рекомендует разработчикам ПО двигаться именно в этом направлении, т. е. внедрять меры по разработке безопасного ПО на всех этапах жизненного цикла. Базовый набор мер, которые необходимо реализовать разработчикам в процессе создания и дальнейшей поддержке своего продукта, представлены в стандарте [1].

Применение мер по безопасной разработке на этапах жизненного цикла ПО

Типовая модель жизненного цикла ПО состоит из работ, последовательно или циклически сменяющих друг друга. Модель начинается с формулировки замысла (идеи) или концепции ПО, продолжается работами по применению методов системной и программной инженерии, работами по эксплуатации, сопровождению и поддержке продукта и заканчивается снятием его с эксплуатации (утилизацией) [2].

Модель жизненного цикла ПО приводится во многих стандартах [1–3], однако ее описание практически не отличается, как и само деление на стадии.

Далее рассмотрим каждый этап жизненного цикла ПО в разрезе рекомендуемых к применению мер по безопасной разработке, и приведем типовые уязвимости, на исключение эксплуатации которых направлены данные меры. В качестве практического примера приведем реальные аспекты адаптации и внедрения принципов безопасной разработки в рамках масштабного проекта ФГУП «РФЯЦ-ВНИИЭФ» по созданию комплекса программ в защищенном исполнении «Система полного жизненного цикла изделий «Цифровое предприятие» (СПЖЦ ЦП).

1. Анализ требований

Во время проведения данной работы устанавливаются требования к разрабатываемому ПО, проводится анализ требований на корректность, определяются приоритеты реализации. Итогом данной фазы должны стать документально оформленные формулировки общих потребностей пользователей с учетом таких факторов, как стоимость, критичность,

реализуемость планируемой разработки. Иначе говоря, на данном этапе формируется техническое задание (ТЗ) [3, 4].

Для реализации меры по разработке безопасного ПО на данном этапе жизненного цикла разработчик должен дополнительно определить требования по безопасности, предъявляемые к разрабатываемому ПО. Источниками для формирования таких требований могут выступать законы, нормативные правовые акты, отраслевые стандарты, требования пользователей, сценарии применения ПО [1]. Например, могут быть определены требования к обеспечению идентификации и аутентификации, к обеспечению защиты от несанкционированного доступа (НСД), к обеспечению регистрации событий. Требования по безопасности должны быть документально оформлены – включены в ТЗ.

Реализация данной меры позволить на следующем этапе выстроить верную концепцию проектирования архитектуры ПО, что позволит избежать большого класса уязвимостей.

В рамках проекта по созданию СПЖЦ ЦП постановка функциональных требований безопасности к компонентам осуществлялась на основе [9], а также ряда руководящих документов, национальных и международных стандартов в области защиты информации. Определена и документально зафиксирована выборка функциональных механизмов, актуальных для заданного класса продуктов. Для постановки задачи на разработку требования к функциональности изделий сформулированы ТЗ на каждый программный модуль (компонент). Помимо требований к разрабатываемым продуктам определены требования к сопутствующим процессам разработки, тестирования, производства и поддержки, требования к документированию и управлению конфигурацией.

2. Проектирование архитектуры ПО

На данном этапе происходит разработка проекта архитектуры ПО, описываются составные части, определяются внутренние и внешние интерфейсы, предполагаемые потоки данных. Другими словами, разработчик описывает верхний уровень структуры ПО и идентифицирует его компоненты.

Для разработки безопасного ПО разработчику дополнительно следует выполнить моделирование угроз безопасности информации, которые могут возникнуть вследствие применения ПО. На основании полученных результатов моделирования угроз могут быть уточнены требования по безопасности, а также проект архитектуры ПО [1–4].

Данная мера позволит исключить потенциальные угрозы безопасности, а также позволит сформировать исходные данные для тестирования ПО.

Первичное моделирование угроз безопасности информации для СПЖЦ ЦП выявило ряд потенциальных угроз, в том числе связанных с возможным нарушением политик управления доступом при взаимодействии отдельных компонентов ПО и среды функционирования. Это позволило уже на этапе проектирования архитектуры заложить отдельные требования к интерфейсам, интеграции и организации потоков данных, что позволяет включить в разработку программных модулей задачи, нацеленные на минимизацию и устранение данных угроз. Далее процессы моделирования угроз безопасности информации систематизированы таким образом, чтобы любые изменения, вносимые в интерфейсы и структуру целевых версий ПО, проходили соответствующую оценку вероятности возникновения новых потенциальных угроз безопасности с целью обеспечения оперативного реагирования со стороны разработчиков.

3. Конструирование и комплексирование ПО

Конструирование заключается в создании исполняемых программных блоков ПО и баз данных, определении процедуры тестирования блоков ПО и данных для тестирования. Комплексирование ПО – в объединении программных блоков и компонентов, создании интегрированных программных элементов [14].

На данном этапе разработчик должен идентифицировать каждое инструментальное средство, используемое при разработке ПО, и определить его настройки. Также необходимо выбрать и использовать порядок оформления исходного кода ПО. Исходный код разработанных программных модулей и компонентов, в том числе заимствованных у сторонних разработчиков, подвергается экспертизе и автоматизированному статическому анализу.

Результатом внедрения таких мер будет являться перечень потенциально уязвимых конструкций в исходном коде ПО, устранение которых значительно повысит уровень защищенности ПО и сократит количество векторов атак потенциальных нарушителей при эксплуатации ПО.

Разработка программных модулей СПЖЦ ЦП осуществляется как силами РФЯЦ-ВНИИЭФ, так и с привлечением сторонних организаций. Во исполнение мер защиты информации на этапе разработки обозначены правила документирования и комментирования создаваемого программного кода, определены требования к используемым языкам и средствам разработки. Каждая версия программного модуля передается специалистам ИБ для проведения статического анализа и экспертизы исходного кода. Результаты анализа фиксируются в протоколах и передаются разработчикам для устранения. В настоящее время ведется документирование всех процедур, связанных с созданием и экспертизой исходного кода.

4. Квалификационное тестирование ПО

Целью квалификационного тестирования является подтверждение того, что реализация каждого механизма ПО соответствует предъявляемым требованиям, в том числе требованиям безопасности информации, и ПО готово к поставке. Тестирование ПО должно осуществляться в соответствии с утвержденными программами и методиками испытаний, результаты подлежат протоколированию [3].

Для соответствия ПО в части внедряемых мер безопасной разработки, а также с целью выявления несоответствий требованиям безопасности и выявле-

нию уязвимостей на данном этапе разработчик должен провести функциональное тестирование ПО, тестирование на проникновение, динамический анализ кода ПО, фаззинг-тестирование. При выявлении уязвимостей или несоответствий по итогам проведенного тестирования ПО разработчики должны устранить все неисправности путем доработки ПО.

Проведение разных видов тестирования подтверждает, что ПО соответствует идентифицированным требованиям по безопасности, установленным на этапе анализа требований, а также не содержит уязвимостей [16].

В отношении программных модулей СПЖЦ ЦП квалификационное тестирование можно условно разделить на два направления.

Первое касается проверки основного функционала ПО, механизмов направленных на выполнение целевых задач. Данный вид тестирования проводится для каждой версии каждого программного модуля.

Второе направление охватывает вопросы обеспечения защиты информации: это фаззинг-тестирование интерфейсов, выявление и оценка скрытых каналов, динамический анализ, проверка корректности реализации механизмов безопасности информации. Данный блок испытаний проводится в отношении целевых версий программных модулей, представляющих собой законченные проекты, последовательно реализуемых в соответствии с дорожной картой Программы этапов создания ПО. В частности, ряд программных модулей СПЖЦ ЦП выходит на такие этапы уже в 2020 г.

5. Инсталляция и поддержка приемки ПО

Данный этап заключается в установке ПО, удовлетворяющего требованиям ТЗ, в целевую среду применения. Иными словами, происходит поставка ПО, ввод в действие и его наладка в соответствии с принятыми в организации политиками и процедурами в отношении процесса инсталляции ПО [17].

Мерами по разработке безопасного ПО в данном случае будут являться обеспечение защиты ПО от угроз, связанных с нарушением целостности, в процессе его передачи пользователю, и поставка пользователю эксплуатационных документов. Для этого могут применяться средства контрольного суммирования, пломбирование упаковки и другие. Эксплуатационные документы в свою очередь должны содержать перечень и эталонные значения конфигурационных параметров ПО.

Внедрение описанных мер позволит исключить возможность поставки модифицированного ПО, то есть с намеренно внесенными уязвимостями или закладками, а также обеспечит безопасную настройку ПО при его инсталляции и первом запуске [1–4].

Для организации производства и технического контроля разрабатываемых программных модулей СПЖЦ ЦП в Институте цифровых технологий РФЯЦ-ВНИИЭФ выделена структурная единица, ведется описание процессов технического контроля, упаковки и маркировки изделий с учетом требований и рекомендаций в области защиты информации. Для каждого поставляемого продукта будет определен комплект поставки, включающий всю необходимую эксплуатационную документацию. В зависимости от комплектности изделия и способа поставки будет определен ряд контролируемых характеристик, которые позволят пользователю подтвердить целостность и неизменность полученного продукта.

6. Эксплуатация ПО

Данный этап включает в себя эксплуатацию ПО в предназначенной для него среде функционирования и обеспечение его технической поддержки, включая работу в режиме диспетчерской связи, до момента снятия ПО с эксплуатации.

Для того, чтобы внедрить меры по безопасной разработке [1] разработчик должен на данном этапе проводить периодическое отслеживание и исправление обнаруженных ошибок и уязвимостей ПО в соответствии с разработанным регламентом. Доведение до пользователей информации об ошибках и уязвимостях ПО должно быть задокументировано [1–4].

Данная мера позволит своевременно обнаруживать ошибки и все виды уязвимостей ПО, а также оперативно их устранять.

Для проекта СПЖЦ ЦП были разработаны документы, охватывающие все возможные варианты обнаружения и устранения ошибок. Учтены, как внешние источники информации, в лице будущих пользователей ПО, государственных регуляторов и независимых исследователей, так и внутренние процедуры периодического тестирования и анализа уязвимостей. Все процессы задокументированы и нацелены на всестороннюю поддержку пользователей и поддержание высокого уровня защищенности ПО в процессе его эксплуатации.

7. Управление документацией и конфигурацией программы

Данный этап не выделяется в стадиях жизненного цикла ПО, т. к. документация может разрабатываться на всех этапах за исключением поставки ПО, его эксплуатации и поддержки. На этих этапах документация может уточняться в соответствии с какими-либо изменениями, доработками.

Тем не менее, существуют меры по разработке безопасного ПО, которые необходимо внедрять разработчикам в процессе разработки документации. К ним относятся определение элементов конфигурации ПО, разработка способа маркировки каждой версии ПО, контроль целостности элементов конфигурации. В область действия системы управления конфигурацией могут входить: дистрибутив ПО, программные и эксплуатационные документы, исходный код программы, модули ПО, инструментальные средства, информация по обновлению ПО [1–4].

Перечисленные меры позволят обеспечить целостность элементов конфигурации ПО, что исключит возможность подмены отдельных составляющих ПО с целью внедрения уязвимостей.

Данные меры также нашли свое отражение на проекте СПЖЦ ЦП. Перечень контролируемых элементов конфигурации включает в себя, как файлы

дистрибутива, библиотек, компонентов, исходных кодов и документации, так и сведения об инструментальных средствах разработки и о выявленных на различных этапах уязвимостях. В среде разработки используются инструменты контроля версий и изменений, определен порядок резервного копирования. Все процессы системы управления конфигурацией документируются и учитывают особенности разработки и поддержки каждого программного модуля.

8. Управление инфраструктурой среды разработки ПО

Этот этап условно можно отнести к стадии разработки ПО, но в [1] он выделяется отдельно. Из самого названия становится ясно, что этап связан с обеспечением безопасности на всем периоде разработки ПО. Это действительно так. Разработчик должен защитить он НСД элементы конфигурации, которые были определены им ранее, должен совершать действия по резервному копированию значимых элементов конфигурации и регистрировать события, связанные с фактами изменения элементов конфигурации ПО.

Для реализации данного требования разработчиками СПЖЦ ЦП были определены элементы конфигурации, которые должны подлежать защите от угроз безопасности информации, связанных с нарушением целостности, доступности и конфиденциальности, и элементы конфигурации, подлежащие резервному копированию. На каждом этапе разработки и для каждого сегмента разработки (автоматизированные сети и рабочие места участников разработки) определен ряд технических и организационных мер, направленных на защиту от НСД. События, связанные с фактами изменения элементов конфигурации фиксируются в соответствующем журнале регистрации [1–4, 10].

9. Управление людскими ресурсами

Чтобы оказывать соответствующую техническую поддержку, а также понимать всю значимость мер по безопасной разработке ПО, разработчикам необходимо проходить периодическое обучение. В учебную программу могут входить курсы по моделированию угроз безопасности информации, экспертизы исходного кода ПО, тестирования на проникновение, статического и динамического анализов.

Обучение разработчиков помогает повысить уровень их компетенций, развить и поддержать навыки в области разработки безопасного ПО, что влечет за собой более ответственный и компетентный подход к реализации мер стандарта [1] на каждом этапе жизненного цикла ПО.

ФГУП «РФЯЦ-ВНИИЭФ» периодически проводит обучение разработчиков СПЖЦ ЦП, в том числе с привлечением сторонних организаций. Разработчики непрерывно повышают свою квалификацию, проходят обучение в известных учебных центрах, включая Институт системного программирования им. В. П. Иванникова Российской академии наук. Для развития компетенций своих сотрудников и с целью привлечения компетентного сообщества в разработку СПЖЦ ЦП была создана среда коллективной разработки (СКР). СКР включает в себя территориально-распределенные сети и сервера, средства разработки и резервного копирования.

Заключение

Создание (разработка) ПО, особенно защищенного ПО, является непростым процессом. Помимо реализации всех требований пользователей в части целевого функционала, необходимо глубоко проработать вопросы обеспечения безопасности информации, которая будет обрабатываться с помощью данного ПО, иначе компания-разработчик не сможет выйти на мировой рынок и составлять конкуренцию своим оппонентам. Внедрение процессов безопасной разработки ПО безусловно накладывает дополнительные обязательства, требует вливания дополнительных объемов финансовых, трудовых и временных ресурсов. Однако в современных реалиях это важный и необходимый шаг, на который должен решиться каждый разработчик [10–14].

Организация и внедрение процессов безопасной разработки ПО приведет к устранению широкого класса уязвимостей, недоработок и ошибок ПО, что позволит говорить о доверенном продукте. С каждым этапом жизненного цикла ПО уровень его защищенности будет только повышаться. Это утверждение подкреплено не только лучшими мировыми практиками, но и собственным опытом.

Масштабный проект по разработке СПЖЦ ЦП показал действенность внедрения методов и мер по безопасной разработке. Уже сейчас можно оценить плоды внедренных подходов на примере огромного пласта неточностей, ошибок и уязвимостей программного кода, выявленных и устраненных на ранних этапах разработки базовых версий программных модулей, что безусловно повысило уровень защищенности разрабатываемых продуктов, и, что в дальнейшем поможет обеспечить положительное прохождение процедуры сертификации, получения сертификата соответствия и гарантировать появление на российском рынке уникального доверенного мощного комплекса, решающего широкий спектр задач.

Список литературы

1. ГОСТ Р 56939–2016. Защита информации. Разработка безопасного программного обеспечения. Общие требования.

2. ГОСТ Р ИСО/МЭК ТО 15271–2002. Информационная технология. Руководство по применению ГОСТ Р ИСО/МЭК 12207 Процессы жизненного цикла программных средств.

3. ГОСТ Р ИСО/МЭК 12207–2010. Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств.

4. ГОСТ Р 56546–2015. Защита информации. Уязвимости информационных систем. Классификация уязвимостей информационных систем.

5. ГОСТ Р 58142–2018. Национальный стандарт Российской Федерации. Информационная технология. Методы и средства обеспечения безопасности. Детализация анализа уязвимостей программного обеспечения в соответствии с ГОСТ Р ИСО/МЭК 15408 и ГОСТ Р ИСО/МЭК 18045. Часть 1. Использование доступных источников для идентификации потенциальных уязвимостей.

6. ГОСТ Р 50.1.053–2005. Информационные технологии. Основные термины и определения в области технической защиты информации.

7. ГОСТ Р 53113.2-2009. Защита информационных технологий и автоматизированных систем от угроз информационной безопасности, реализуемых с использованием скрытых каналов. Часть 2. Рекомендации по организации защиты информации, информационных технологий и автоматизированных систем от атак с использованием скрытых каналов.

8. ГОСТ Р 50922–2006. Национальный стандарт Российской Федерации. Защита информации. Основные термины и определения.

9. ГОСТ Р ИСО/МЭК 15408–2-2013. Национальный стандарт Российской Федерации. Информационная технология. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 2. Функциональные компоненты безопасности.

10. Барабанов А. В. Нормативные вопросы безопасного производства программ // Правовая информатика. 2013. № 2. С. 49–53.

11. Барабанов А. В. Стандартизация процессов разработки безопасных программных средств // Вопросы кибербезопасности. 2013. № 1(1). С. 37–41.

12. Марков А. С., Фадин А. А. Систематика уязвимостей и дефектов безопасности программных ресурсов //Защита информации. INSIDE. 2013. № 3, С. 2–7.

13. Иванько А. Ф., Иванько М. А., Шанина А. А. Информационная безопасность вчера и сегодня // Молодой ученый. 2017. № 51 (185). С. 25–30.

14. Дудкина И. А. Технологии и методы обеспечения комплексной защиты информации // Молодой ученый. 2016. № 16 (120). С. 37–39.

15. Репин М. Е., Афанасьев А. Ю. Преступления в сфере компьютерной информации: проблемы выявления и раскрытия // Молодой ученый. 2015. № 15 (95). С. 460–463.

16. Ортыков А. У. Обеспечение информационной безопасности предприятия от несанкционированного доступа / III Международная научная конференция. Технические науки: традиции и инновации // Материалы конференции. Казань: 2018. С. 22–24.

17. Лободина А. С., Ермолаева В. В. Информационная безопасность // Молодой ученый. 2017. № 17 (151). С. 17–20.

18. Информационный портал по вопросам безопасности в сети интернет [Электронный ресурс] https://safe-surf.ru (дата обращения: 16.08.2020).

19. Сайт компании Positive Technologies [Электронный ресурс] https://www.ptsecurity.com (дата обращения: 16.08.2020).

20. Сайт компании XB Software [Электронный ресурс] https://xbsoftware.ru/ (дата обращения: 16.08.2020).