23. БЕЗОПАСНОСТЬ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

23.1 Введение в защиту ПО

Безопасность программного обеспечения *(ПО) в широком смысле является свойством данного ПО функционировать без проявления различных негативных последствий для конкретной компьютерной системы.*

Под уровнем безопасности ПО *понимается вероятность того, что при заданных условиях в процессе его эксплуатации будет получен функционально пригодный результат.*

Причины, приводящие к функционально непригодному результату могут быть разными: сбои компьютерных систем, ошибки программистов и операторов, дефекты в ПО. При этом дефекты принято рассматривать двух типов: преднамеренные и непреднамеренные. Первые являются, как правило, результатом злоумышленных действий, вторые - ошибочных действий человека.

При исследовании проблем защиты ПО от преднамеренных дефектов неизбежна постановка следующих вопросов:

- кто потенциально может осуществить практическое внедрение программных дефектов деструктивного воздействия в исполняемый программный код;
- каковы возможные мотивы действий субъекта, осуществляющего разработку таких дефектов;
- как можно идентифицировать наличие программного дефекта;
- как можно отличить преднамеренный программный дефект от программной ошибки;
- каковы наиболее вероятные последствия активизации деструктивных программных средств при эксплуатации компьютерных систем (КС).

При ответе на первый вопрос следует отметить, что это: непосредственные разработчики алгоритмов и программ для компьютерных систем. Они хорошо знакомы с технологией разработки программных средств, имеют опыт разработки алгоритмов и программ для конкретных прикладных систем, знают тонкости существующей технологии отработки и испытаний программных компонентов и представляют особенности эксплуатации и целевого применения разрабатываемой КС.

Отметим, что алгоритмические и программные закладки могут быть реализованы в составе программного компонента вследствие следующих факторов:

- в результате инициативных злоумышленных действий непосредственных разработчиков алгоритмов и программ;

- в результате штатной деятельности специальных служб и организаций, а также отдельных злоумышленников;
- в результате применения инструментальных средств проектирования ПО, несущих вредоносное свойство автоматической генерации деструктивных программных средств.

Правомерно утверждать, что вредоносные программы, в отличие от широко применяемых электронных закладок, являются более изощренными объектами, обладающими большей скрытностью и эффективностью применения.

23.2 Угрозы безопасности ПО

Угрозы безопасности информации и программного обеспечения компьютерных систем возникают как в процессе их эксплуатации, так и при создании этих систем, что особенно характерно для процесса разработки ПО, баз данных и других информационных компонентов КС.

Наиболее уязвимы с точки зрения защищенности информационных ресурсов являются так называемые ***критические компьютерные системы.***

Критические компьютерные системы *- сложные компьютеризированные организационно-технические и технические системы, блокировка или нарушение функционирования которых потенциально приводит к потере устойчивости организационных систем государственного управления и контроля, утрате обороноспособности государства, разрушению системы финансового обращения, дезорганизации систем энергетического и коммуникационно - транспортного обеспечения государства, глобальным экологическим и техногенным катастрофам.*

При решении проблемы повышения уровня защищенности информационных ресурсов КС необходимо исходить из того, что *наиболее вероятным информационным объектом воздействия будет выступать программное обеспечение, составляющее основу комплекса средств получения, семантической переработки, распределения и хранения данных, используемых при эксплуатации критических систем.*

Программные средства деструктивного воздействия по своей природе носят, вредоносный характер, а последствия их активизации и применения могут привести к значительному или даже непоправимому ущербу. *Такие вредоносные программы будем называть* ***разрушающими программными средствами (РПС),*** *а их обобщенная* ***классификация*** *может выглядеть следующим образом:*

- ***компьютерные вирусы*** - программы, способные размножаться, прикрепляться к другим программам, передаваться по линиям связи

и сетям передачи данных, проникать в электронные телефонные станции и системы управления и выводить их из строя;

- ***программные закладки*** - *программные компоненты, заранее внедряемые в компьютерные системы, которые по сигналу или в установленное время приводятся в действие, уничтожая или искажая информацию, или дезорганизуя работу программно-технических средств;*
- ***способы и средства, позволяющие внедрять компьютерные вирусы и программные закладки*** *в компьютерные системы и управлять ими на расстоянии*.

*Под **алгоритмической закладкой** будем понимать преднамеренное завуалированное искажение какой-либо части алгоритма решения задачи, либо построение его таким образом, что в результате конечной программной реализации этого алгоритма в составе программного компонента или комплекса программ, последние будут иметь ограничения на выполнение требуемых функций, заданных спецификацией, или вовсе их не выполнять при определенных условиях протекания вычислительного процесса*, задаваемого семантикой перерабатываемых программой данных. Кроме того, возможно появление у программного компонента функций, не предусмотренных прямо или косвенно спецификацией, и которые могут быть выполнены при строго определенных условиях протекания вычислительного процесса.

*Под **программной закладкой** будем понимать совокупность операторов и (или) операндов, преднамеренно в завуалированной форме включаемую в состав выполняемого кода программного компонента на любом этапе его разработки*. Программная закладка реализует определенный несанкционированный алгоритм с целью ограничения или блокирования выполнения программным компонентом требуемых функций при определенных условиях протекания вычислительного процесса, задаваемого семантикой перерабатываемых программным компонентом данных, либо с целью снабжения программного компонента не предусмотренными спецификацией функциями, которые могут быть выполнены при строго определенных условиях протекания вычислительного процесса.

Примеры уязвимостей ПО

Команда из четырех человек, работающих в разных университетах США, в 2008 году разработала **технологию автоматической генерации кода атаки на такую уязвимость в ПО, которая заранее неизвестна, а вычисляется путем сличения исходной и пропатченной версий программы**. Иначе говоря, инструкции для создания нового вредоносного

кода предоставляет, по сути, сама программная заплатка, выпущенная с целью латания очередной дыры.

Разработанная технология **APEG (Automatic Patch-based Exploit Generation)** позволяет за время от нескольких секунд до нескольких минут сгенерировать код атаки для большинства типов программных уязвимостей. Алгоритм APEG работает как доказательство корректности системы, проводимое в обратную сторону. Сначала выявляются различия в исполняемых кодах программы до и после применения заплатки, а затем по ее коду анализируется, для чего она предназначена. Патчи безопасности обычно содержат тест, который определенным образом ограничивает допустимые значения на входе системы, но существует процедура позволяющая пройти по коду и автоматически выявить набор входов, которые отлавливаются тестами нового патча. Когда это сделано, применяется специальный набор правил-эвристик для точной локализации места уязвимости, затем генерируется несколько вариантов кодов, потенциально способных эксплуатировать данную уязвимость, а тестовые испытания устанавливают, какой из кодов реально срабатывает.

По мнению разработчиков, это означает, что если корпорация Microsoft существенно не изменит способ распространения патчей среди клиентов, то последствия могут оказаться тяжелейшими. Ведь злоумышленники, заполучив в руки систему типа APEG, могут обнаружить уязвимость по свежевыпущенному патчу и провести атаку до того, как этот самый патч будет установлен на атакуемую машину.

Сотрудники Иллинойского университета (Урбана-Шампань) на конференции семинара по крупномасштабным и новым компьютерным угрозам (USENIX workshop on Large-Scale Exploits and Emergent Threats [LEET]) в 2008 году представили на удивление **эффективный подход к добавлению аппаратных закладок в компьютеры общего назначения**. Исследователи показали, что внесения в схему процессора совсем небольшого (одна-две тысячи) числа элементов достаточно для обеспечения широкого спектра дистанционных атак, которые невозможно выявить или предотвратить с помощью традиционных софтверных подходов к безопасности. Правда, для проведения подобных атак требуется фундаментально скомпрометировать компьютеры на этапе их создания или сборки. Это вполне по силам государственным спецслужбам.

Технически это выглядит так. Скрытые в процессоре вредоносные схемы обеспечивают атакующую сторону невидимым внутренним плацдармом для атак. Поскольку такие схемы занимают уровень, находящийся ниже стека программ, они способны обходить все традиционные техники защиты.

В работе представлена общая конструкция и конкретные формы реализации так называемых IMPs (Illinois Malicious Processors, «иллинойских вредоносных процессоров»). Показано, что даже с учетом жестких

ограничений по месту, его все равно достаточно для планирования разнообразных типов атак, а не одной узконаправленной. Такая гибкость схемы позволила разработчикам продемонстрировать две конкретные конструкции и реализовать их практически в конкретной системе FPGA-чипа, то есть процессора с перепрограммируемой логикой.

Вот примеры, подтверждающие общую концепцию.

Эскалация привилегий. Используя механизм доступа к памяти, реализован вредоносный сервис, поднимающий привилегии пользовательского процесса до высшего (root) уровня. При выполнении такой атаки программа эскалации привилегий использует аппаратную закладку в процессоре для отключения защиты привилегированных областей памяти. Он позволяет напрямую нарушать все предположения ОС относительно обеспечиваемой защиты памяти.

Входной бэкдор. Используя специально разработанный механизм теневого режима, разработчики реализовали вредоносный сервис, служащий постоянным «черным ходом» в систему. Чтобы начать атаку захвата, злоумышленник посылает сетевой пакет в систему жертвы, где ОС первым делом инспектирует этот пакет, проверяя контрольную сумму UDP. Сам акт проверки пакета (необходимый для принятия решения о том, следует ли его отвергнуть) запускает троянскую закладку в железе, а вредоносная программа интерпретирует содержимое пакета как новый код прошивки, который невидимо загружается в процессор. Операционная же система тем временем отбрасывает непрошеный пакет и продолжает работу, не заметив атаки.

Код прошивки, реализующий теневой режим, отслеживает login-приложение для входа в систему. И когда некто пытается войти с особым, заранее известным закладке паролем, та подменяет значение функции проверки пароля на «правильный» и тем самым гарантирует доступ в систему любому, кто знает хитрость. Чтобы скрыть следы атаки, сразу после успешной попытки логина прошивка сама себя выгружает и отключает теневой режим, возвращая системе все ресурсы процессора. Таким образом, послав сетевой UDP-пакет и тут же войдя в систему, злоумышленник может сократить время работы теневого режима до минимума. Если же система жертвы не имеет выхода в сеть, то для включения закладки-бэкдора можно использовать похожий механизм на основе внешнего накопителя. Например, в USB-модуле флэш-памяти для этого подходит самый первый блок, необходимый для идентификации типа файловой системы. Механизм теневого режима увеличивает количество логических гейтов схемы всего на 0,08%, давая при этом неограниченный доступ к компьютеру без опоры на какие-либо программные уязвимости.

Похищение паролей. С помощью того же механизма теневого режима можно реализовать сервис, ворующий пароли доступа у легитимных пользователей системы. Главная трудность здесь - отыскание паролей в гигантских массивах случайных данных. Но и эта задача вполне разрешима,

коль скоро в символьных строках кода, относящегося к записи и считыванию паролей, присутствует слово Password. В развитие этой же темы исследователи продемонстрировали и два существенно разных способа для скрытного слива похищенных паролей в сеть - как на уровне ОС, так и на уровне прямой модификации пакетов.

Подводя итог, иллинойские исследователи без ложной скромности отмечают, что им удалось заложить фундаментальные основы конструирования процессоров с аппаратными закладками, способными обеспечивать весьма сложные и продвинутые атаки для тех, кто владеет секретами конструкции. Сделано же это, по словам разработчиков, дабы продемонстрировать, что при нынешней организации поставок микросхем заказчикам имеются все предпосылки для злоупотреблений. То есть заинтересованные структуры, обладающие компетентными специалистами и надлежащими ресурсами, вполне способны разрабатывать и внедрять вредоносные микросхемы с аппаратными закладками.

Действия алгоритмических и программных закладок условно можно разделить на три класса:

- *изменение функционирования вычислительной системы (сети),*
- *несанкционированное считывание информации,*
- *несанкционированная модификация информации, вплоть до ее уничтожения.*

Следует отметить, что указанные классы воздействий могут пересекаться.

Класс воздействий **«изменение функционирования вычислительной системы»** направлен на:

- уменьшение скорости работы вычислительной системы (сети);
- частичное или полное блокирование работы системы (сети);
- имитация физических (аппаратурных) сбоев работы вычислительных средств и периферийных устройств;
- переадресация сообщений;
- обход программно-аппаратных средств криптографического преобразования информации;
- обеспечение доступа в систему с непредусмотренных периферийных устройств.

Несанкционированное считывание информации, осуществляемое в автоматизированных системах, направлено на:

- считывание паролей и их отождествление с конкретными пользователями;
- получение секретной информации;
- идентификацию информации, запрашиваемой пользователями;
- подмену паролей с целью доступа к информации;

- контроль активности абонентов сети для получения косвенной информации о взаимодействии пользователей и характере информации, которой обмениваются абоненты сети.

Несанкционированная модификация информации является наиболее опасной разновидностью воздействий программных закладок, поскольку приводит к наиболее опасным последствиям. В этом классе воздействий можно выделить следующие:

- разрушение данных и кодов исполняемых программ внесение тонких, трудно обнаруживаемых изменений в информационные массивы;
- внедрение программных закладок в другие программы и подпрограммы (вирусный механизм воздействий);
- искажение или уничтожение собственной информации сервера и тем самым нарушение работы сети;
- модификация пакетов сообщений.

С точки зрения времени внесения программных закладок в программы их можно разделить на две категории:

1. **априорные**, то есть закладки, внесенные при разработке ПО (или «врожденные»)
2. **апостериорные**, закладки, внесенные при испытаниях, эксплуатации или модернизации ПО (или «приобретенные»).

Хотя последняя разновидность закладок и относятся больше к проблеме обеспечения эксплуатационной, а не технологической безопасности ПО, однако методы тестирования программных комплексов, вероятностные методы расчета наличия программных дефектов и методы оценивания уровня безопасности ПО могут в значительной мере пересекаться и дополнять друг друга. Тем более что действие программной закладки после того как она была внесена в ПО либо на этапе разработки, либо на последующих этапах жизненного цикла ПО, практически не будет ничем не отличаться.

23.3 Разрушающие программные средства

Необходимым условием для отнесения программы к классу разрушающих программных средств *является наличие в ней процедуры нападения, которую можно определить как процедуру нарушения целостности вычислительной среды,* поскольку объектом нападения РПС всегда выступает элемент этой среды.

При этом необходимо учитывать два фактора:

1. любая прикладная программа, не относящаяся к числу РПС, потенциально может содержать в себе алгоритмические ошибки,

появление которых при ее функционировании приведет к непреднамеренному разрушению элементов вычислительной среды; 2. любая прикладная или сервисная программа, ориентированная на работу с конкретными входными данными может нанести непреднамеренный ущерб элементам операционной или вычислительной среды в случае, когда входные данные либо отсутствуют, либо не соответствуют заданным форматам их ввода в программу.

Для устранения указанной неопределенности по отношению к испытываемым программам следует исходить из предположения, что процедура нарушения целостности вычислительной среды введена в состав ПО умышленно.

Кроме условия необходимости, целесообразно ввести ***условия достаточности****, которые обеспечат возможность описания РПС различных классов:*

- *достаточным условием для отнесения РПС к классу компьютерных* ***вирусов*** *является наличие в его составе процедуры* ***саморепродукции****;*
- *достаточным условием для отнесения РПС к классу* ***средств несанкционированного доступа*** *являются наличие в его составе* ***процедуры преодоления защиты и отсутствия процедуры саморепродукции****;*
- *достаточным условием для отнесения РПС к классу* ***программных закладок*** *является* ***отсутствие*** *в его составе процедур саморепродукции и преодоления защиты.*

Предполагается наличие в РПС следующего набора возможных функциональных элементов:

- *процедуры захвата (получения) управления;*
- *процедуры самомодификации («мутации»);*
- *процедуры порождения (синтеза);*
- *процедуры маскировки (шифрования).*

Этих элементов достаточно для построения обобщенной концептуальной модели РПС, которая отражает возможную структуру (на семантическом уровне) основных классов РПС.

23.4 Модель угроз и принципы обеспечения безопасности ПО

Модель угроз технологической безопасности ПО должна представлять собой официально принятый нормативный документ, которым должен руководствоваться заказчики и разработчики программных комплексов.

Модель угроз должна включать:

полный реестр типов возможных программных закладок;

описание наиболее технологически уязвимых мест компьютерных систем (с точки зрения важности и наличия условий для скрытого внедрения программных закладок);
описание мест и технологические карты разработки программных средств, а также критических этапов, при которых наиболее вероятно скрытое внедрение программных закладок;
реконструкцию замысла структур, имеющих своей целью внедрение в ПО заданного типа (класса, вида) программных закладок диверсионного типа;
психологический портрет потенциального диверсанта в компьютерных системах .

На базе утвержденной модели угроз технологической безопасности должна разрабатываться прикладная модель угроз безопасности для каждого конкретного компонента защищаемого комплекса средств автоматизации КС. В основе этой разработки должна лежать схема угроз представленная на рис. 23.1.

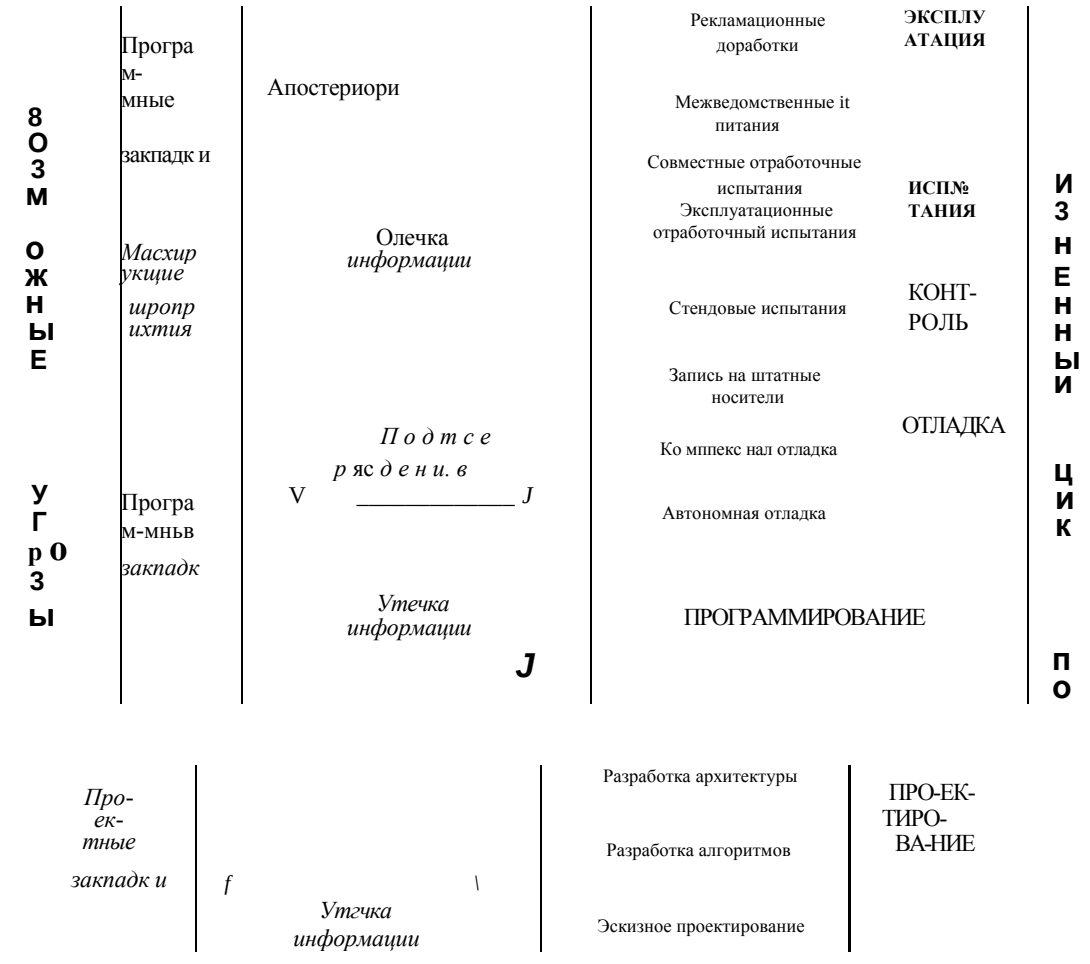

Рис. 23.1 - Прикладная модель угроз безопасности

Наполнение модели технологической безопасности ПО должно включать в себя следующие элементы:

- *матрицу чувствительности КС к «вариациям» ПО (то есть к появлению искажений),*
- *энтропийный портрет ПО (то есть описание «темных» запутанных участков ПО),*
- *реестр камуфлирующих условий для конкретного ПО,*
- *справочные данные о разработчиках*
- *реальный (либо реконструированный) замысел злоумышленников по поражению этого ПО.*

В таблице 23.1 приведен пример указанной типовой модели для сложных программных комплексов.

Таблица 23.1. Пример типовой модели технологической безопасности ПО для сложных программных комплексов

№	**Угрозы рассматриваемые в модели безопасности в соответсвии с <u>жизненным циклом ПО</u>**
1	ПРОЕКТИРОВАНИЕ
1.1	*Проектные решения* - Злоумышленный выбор нерациональных алгоритмов работы Облегчение внесения закладок и затруднение их обнаружения. - Внедрение злоумышленников в коллективы, разрабатывающие наиболее ответственные части ПО.
1.2	*Используемые информационные технологии* - Внедрение злоумышленников, в совершенстве знающих «слабые» места и особенности используемых технологий. Внедрение информационных технологий или их элементов, содержащих программные закладки. - Внедрение неоптимальных информационных технологий.
1.3	*Используемые аппаратно-технические средства* - Поставка вычислительных средств, содержащих программные, аппаратные или программно-аппаратные закладки. - Поставка вычислительных средств с низкими реальными характеристиками. - Поставка вычислительных средств, имеющих высокий уровень экологической опасности. - Задачи коллективов разработчиков и их персональный состав. - Внедрение злоумышленников в коллективы разработчиков программных и аппаратных средств. - Вербовка сотрудников путем подкупа, шантажа и т.п.

2 КОДИРОВАНИЕ

2.1 *Архитектура программной системы, взаимодействие ее с внешней средой и взаимодействие подпрограмм программной системы*

- Доступ к «чужим» подпрограммам и данным.
- Нерациональная организация вычислительного процесса.
- Организация динамически формируемых команд или параллельных вычислительных процессов.
- Организация переадресации команд, запись злоумышленной информации в используемые программной системой или другими программами ячейки памяти.

2.2 *Функции и назначение кодируемой части программной системы, взаимодействие этой части с другими подпрограммами*

- Формирование программной закладки, воздействующей на другие части программной системы.
- Организация замаскированного спускового механизма программной закладки.

  Формирование программной закладки, изменяющей структуру программной системы.

2.3 *Технология записи программного обеспечения и исходных данных*

- Поставка программного обеспечения и технических средств со встроенными дефектами.

3 ОТЛАДКА И ИСПЫТАНИЯ

3.1 *Назначение, функционирование, архитектура программной системы*

- Встраивание программной закладки как в отдельные подпрограммы, так и в управляющую программу программной системы.
- Формирование программной закладки с динамически формируемыми командами.
- Организация переадресации отдельных команд программной системы.

3.2 *Сведения о процессе испытаний (набор тестовых данных, используемые вычислительные средства, подразделения и лица, проводящие испытания, используемые модели*

- Формирование набора тестовых данных, не позволяющих выявить программную закладку.
- Поставка вычислительных средств, содержащих программные, аппаратные или программно-аппаратные закладки.
- Формирование программной закладки, не обнаруживаемой с помощью используемой модели объекта в силу ее неадекватности описываемому объекту.
- Вербовка сотрудников коллектива, проводящих испытания.

4 4.1 КОНТРОЛЬ

Используемые процедуры и методы контроля

- Формирование спускового механизма программной закладки, не включающего ее при контроле на безопасность.
- Маскировка программной закладки путем внесения в программную систему ложных «непреднамеренных» дефектов.
- Формирование программной закладки в ветвях программной системы, не проверяемых при контроле.
- Формирование «вирусных» программ, не позволяющих выявить их внедрение в программную систему путем контрольного суммирования.
- Поставка программного обеспечения и вычислительной техники, содержащих программные, аппаратные и программно-аппаратные закладки.

5 ЭКСПЛУАТАЦИЯ

5.1 *Сведения о персональном составе контролирующего подразделения и испытываемых программных системах*

- Внедрение злоумышленников в контролирующее подразделение.
- Вербовка сотрудников контролирующего подразделения.

5.2 - Сбор информации о испытываемой программной системе.

Сведения об обнаруженных при контроле программных закладках

- Разработка новых программных закладок при доработке программной системы.

5.3 *Сведения об обнаруженных незлоумышленных дефектах и программных закладках*

5.4 *Сведения о доработках программной системы и подразделениях, их осуществляющих*

5.5 *Сведения о среде функционирования программной системы и ее изменениях*

5.6 *Сведения о функционировании программной системы, доступе к ее загрузочному модулю и исходным данным, алгоритмах проверки сохранности программной системы и данных*

23.5 Элементы модели угроз эксплуатационной безопасности ПО

Анализ угроз эксплуатационной безопасности ПО КС позволяет, разделить их на два типа:

1. случайные;
2. преднамеренные:
   - активные,
   - пассивные.

Активные угрозы направлены на изменение технологически обусловленных алгоритмов, программ функциональных преобразований или информации, над которой эти преобразования осуществляются.

Пассивные угрозы ориентированы на нарушение безопасности информационных технологий без реализации таких модификаций.

Вариант общей структуры набора потенциальных угроз безопасности информации и ПО на этапе эксплуатации КС приведен в табл. 23.2.

Рассмотрим основное содержание данной таблицы. Угрозы, носящие случайный характер и связанные с отказами, сбоями аппаратуры, ошибками операторов и т.п. предполагают нарушение заданного собственником информации алгоритма, программы ее обработки или искажение содержания этой информации. Субъективный фактор появления таких угроз обусловлен ограниченной надежностью работы человека и проявляется в виде ошибок (дефектов) в выполнении операций формализации алгоритма функциональных преобразований или описания алгоритма на некотором языке, понятном вычислительной системе.

Угрозы, носящие злоумышленный характер вызваны, как правило, преднамеренным желанием субъекта осуществить несанкционированные изменения с целью нарушения корректного выполнения преобразований, достоверности и целостности данных, которое проявляется в искажениях их содержания или структуры, а также с целью нарушения функционирования технических средств в процессе реализации функциональных преобразований и изменения конструктива вычислительных систем и систем телекоммуникаций.

На основании анализа уязвимых мест и после составления полного перечня угроз для данного конкретного объекта информационной защиты, например, в виде указанной таблицы, необходимо осуществить переход к неформализованному или формализованному описанию модели угроз эксплуатационной безопасности ПО КС. Такая модель, в свою очередь, должна соотноситься (или даже являться составной частью) обобщенной модели обеспечения безопасности информации и ПО объекта защиты.

После окончательного синтеза модели угроз разрабатываются практические рекомендации и методики по ее использованию для конкретного объекта информационной защиты, а также механизмы оценки адекватности модели и реальной информационной ситуации и оценки эффективности ее применения при эксплуатации КС.

Таблица 23.2. Вариант общей структуры набора потенциальных угроз безопасности информации и ПО на этапе эксплуатации КС

	Несанкционированные действия		
	Случайные	**Преднамеренные**	
		Пассивные	**Активные**
	невыявленные ошибки программного обеспечения КС; отказы и сбои технических средств КС; ошибки операторов; неисправность средств шифрования; скачки электропитания на технических средствах; старение носителей информации; разрушение информации под воздействием физических факторов(аварии и т.п.).	маскировка несанкционированных запросов под запросы ОС; обход программ разграничения доступа; чтение конфиденциальных данных из источников информации; подключение к каналам связи с целью получения информации («подслушивание» и/или «ретрансляция»); при анализе трафика использование терминалов и ЭВМ других операторов; намеренный вызов случайных факторов	включение в программы РПС, выполняющих функции нарушения целостности и конфиденциальности информации и ПО; ввод новых программ, выполняющих функции нарушения безопасности ПО; незаконное применение ключей разграничения доступа; обход программ разграничения доступа; вывод из строя подсистемы регистрации и учета; уничтожение ключей шифрования и паролей; подключение к каналам связи с целью модификации, уничтожения, задержки и переупорядочивания данных; вывод из строя элементов физических средств защиты информации КС; намеренный вызов случайных факторов.
	нарушение пропускного режима и режима секретности; естественные потенциальные поля; помехи и т.п.	перехват ЭМИ от технических средств; хищение производственных отходов(распечаток);визуальный канал; подслушивающие устройства; дистанционное фотографирование и т.п	помехи; отключение электропитания; намеренный вызов случайных факторов.

23.6 Основные принципы обеспечения безопасности ПО на различных стадиях его жизненного цикла

В качестве объекта обеспечения технологической и эксплуатационной безопасности ПО рассматривается вся совокупность его компонентов в рамках конкретной КС. В качестве доминирующей должна использоваться стратегия сквозного тотального контроля технологического и эксплуатационного этапов жизненного цикла компонентов ПО.

23.6.1 Обеспечение безопасности при обосновании, планировании работ и проектном анализе ПО

Принципы обеспечения безопасности на этапах обоснования, планирования работ и проектном анализе ПО включают следующие принципы.

Комплексности обеспечения безопасности ПО, предполагающей рассмотрение проблемы безопасности информационно - вычислительных процессов с учетом всех структур КС, возможных каналов утечки информации и несанкционированного доступа к ней, времени и условий их возникновения, комплексного применения организационных и технических мероприятий.

Планируемости применения средств безопасности программ, предполагающей перенос акцента на совместное системное проектирование ПО и средств его безопасности, планирование их использования в предполагаемых условиях эксплуатации.

Обоснованности средств обеспечения безопасности ПО, заключающейся в глубоком научно-обоснованном подходе к принятию проектных решений по оценке степени безопасности, прогнозированию угроз безопасности, всесторонней априорной оценке показателей средств защиты.

Достаточности безопасности программ, отражающей необходимость поиска наиболее эффективных и надежных мер безопасности при одновременной минимизации их стоимости.

Гибкости управления защитой программ, требующей от системы контроля и управления обеспечением информационной безопасности ПО способности к диагностированию, опережающей нейтрализации, оперативному и эффективному устранению возникающих угроз в условиях резких изменений обстановки информационной борьбы.

Заблаговременности разработки средств обеспечения безопасности и контроля производства ПО, заключающейся в предупредительном характере мер обеспечения технологической безопасности работ в интересах недопущения снижения эффективности системы безопасности процесса создания ПО.

Документируемости технологии создания программ, подразумевающей разработку пакета нормативно-технических документов по контролю программных средств на наличие преднамеренных дефектов.

23.6.2 Обеспечение безопасности ПО в процессе его разработки

Принципы обеспечения безопасности ПО в процессе его разработки включают следующие принципы.

Регламентации технологических этапов разработки ПО, включающей упорядоченные фазы промежуточного контроля, спецификацию программных модулей и стандартизацию функций и формата представления данных.

Автоматизации средств контроля управляющих и вычислительных программ на наличие дефектов, создания типовой общей информационной базы алгоритмов, исходных текстов и программных средств, позволяющих выявлять преднамеренные программные дефекты.

Последовательной многоуровневой фильтрации программных модулей в процессе их создания с применением функционального дублирования разработок и поэтапного контроля.

Типизации алгоритмов, программ и средств информационной безопасности, обеспечивающей информационную, технологическую и программную совместимость, на основе максимальной их унификации по всем компонентам и интерфейсам.

Централизованного управления базами данных проектов ПО и администрирование технологии их разработки с жестким разграничением функций, ограничением доступа в соответствии со средствами диагностики, контроля и защиты.

Блокирования несанкционированного доступа соисполнителей и абонентов государственных сетей связи, подключенных к стендам для разработки программ.

Статистического учета и ведения системных журналов о всех процессах разработки ПО с целью контроля технологической безопасности.

Использования только сертифицированных и выбранных в качестве единых инструментальных средств разработки программ для новых технологий обработки информации и перспективных архитектур вычислительных систем.

23.6.3 Обеспечение безопасности ПО на этапах стендовых и приемо-сдаточных испытаний

Принципы обеспечения безопасности ПО на этапах стендовых и приемо-сдаточных испытаний включают принципы.

Тестирования ПО на основе разработки комплексов тестов, параметризуемых на конкретные классы программ с возможностью

функционального и статистического контроля в широком диапазоне изменения входных и выходных данных.

Проведения натурных испытаний программ при экстремальных нагрузках с имитацией воздействия активных дефектов.

Осуществления «фильтрации» программных комплексов с целью выявления возможных преднамеренных дефектов определенного назначения на базе создания моделей угроз и соответствующих сканирующих программных средств.

Разработки и экспериментальной отработки средств верификации программных изделий.

Проведения стендовых испытаний ПО для определения непреднамеренных программных ошибок проектирования и ошибок разработчика, приводящих к невыполнению целевых функций программ, а также выявление потенциально «узких» мест в программных средствах для разрушительного воздействия.

Отработки средств защиты от несанкционированного воздействия нарушителей на ПО.

Сертификации программных изделий АСУ по требованиям безопасности с выпуском сертификата соответствия этого изделия требованиям технического задания.

23.6.4 Обеспечение безопасности при эксплуатации ПО

Принципы обеспечения безопасности при эксплуатации ПО включают следующие принципы.

Сохранения и ограничения доступа к эталонам программных средств, недопущение внесения изменений в них.

Профилактического выборочного тестирования и полного сканирования программных средств на наличие преднамеренных дефектов.

Идентификации ПО на момент ввода его в эксплуатацию в соответствии с предполагаемыми угрозами безопасности ПО и его контроль.

Обеспечения модификации программных изделий во время их эксплуатации путем замены отдельных модулей без изменения общей структуры и связей с другими модулями.

Строгого учета и каталогизации всех сопровождаемых программных средств, а также собираемой, обрабатываемой и хранимой информации.

Статистического анализа информации обо всех процессах, рабочих операциях, отступлениях от режимов штатного функционирования ПО.

Гибкого применения дополнительных средств защиты ПО в случае выявления новых, непрогнозируемых угроз информационной безопасности.

23.7 Методы и средства анализа безопасности ПО

Широко известны различные средства программного обеспечения обнаружения элементов РПС - от простейших антивирусных программ-сканеров до сложных отладчиков и дизассемблеров - анализаторов и именно на базе этих средств и выработался набор методов, которыми осуществляется анализ безопасности ПО.

В целом полный процесс анализа ПО включает в себя три вида анализа:

1. *лексический верификационный анализ;*
2. *синтаксический верификационный анализ;*
3. *семантический анализ программ.*

Каждый из видов анализа представляет собой законченное исследование программ согласно своей специализации.

Лексический верификационный анализ предполагает поиск распознавания и классификацию различных лексем объекта исследования (программа), представленного в исполняемых кодах. При этом лексемами являются сигнатуры. Поиск лексем (сигнатур) реализуется с помощью специальных программ-сканеров. В данном случае осуществляется поиск сигнатур следующих классов:

- сигнатуры вирусов;
- сигнатуры элементов РПС;
- сигнатуры (лексемы) «подозрительных функций»;
- сигнатуры штатных процедур использования системных ресурсов и внешних устройств.

Синтаксический верификационный анализ предполагает поиск, распознавание и классификацию синтаксических структур РПС, а также построение структурно-алгоритмической модели самой программы.

Семантический анализ предполагает исследование программы изучения смысла составляющих ее функций (процедур) в аспекте операционной среды компьютерной системы. В отличие от предыдущих видов анализа, основанных на статическом исследовании, семантический анализ нацелен на изучение динамики программы - ее взаимодействия с окружающей средой.

Методы анализа безопасности программного обеспечения делят на:

1. *Контрольно-испытательные методы анализа,*
2. *Логико-аналитические методы контроля безопасности программ.*

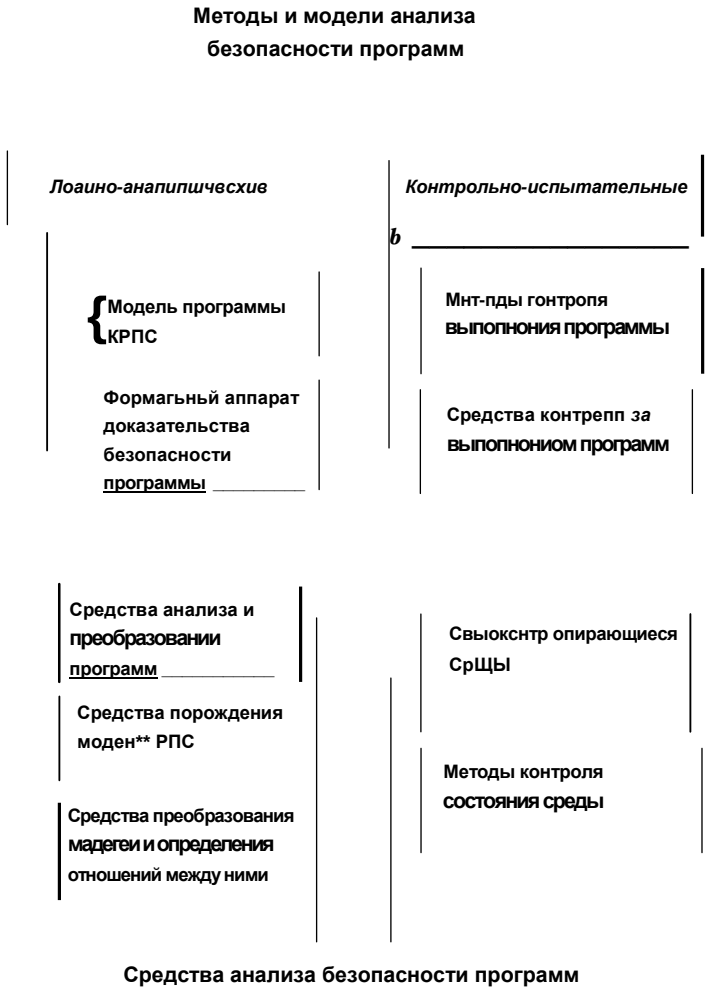

Рис. 23.2 - Методы и средства анализа безопасности ПО

Контрольно-испытательные методы *- это методы, в которых критерием безопасности программы служит факт регистрации в ходе тестирования программы нарушения требований по безопасности, предъявляемых в системе предполагаемого применения исследуемой программы.*

Контрольно-испытательные методы делятся на:

- те, в которых контролируется процесс выполнения программы
- те, в которых отслеживаются изменения в операционной среде, к которым приводит запуск программы. Эти методы наиболее распространены, так как они не требуют формального анализа, позволяют использовать имеющиеся технические и программные средства и быстро ведут к созданию готовых методик.

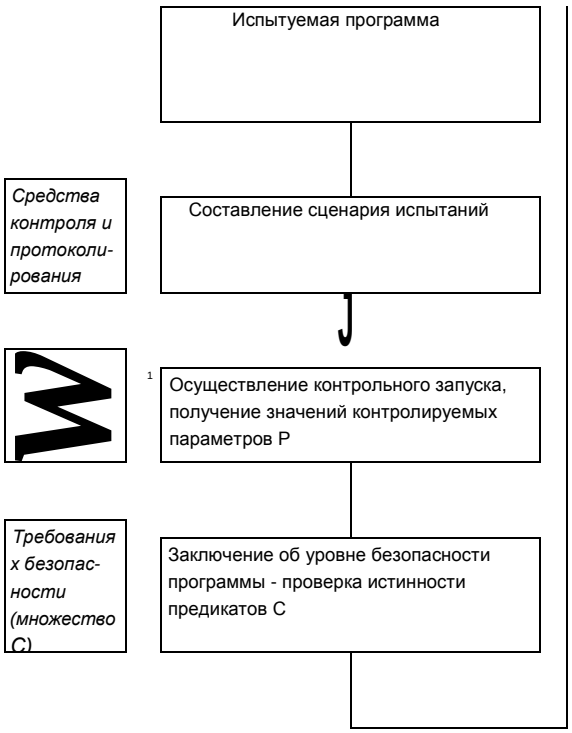

Рис.23.3 - Схема анализа безопасности ПО с помощью контрольно-аналитических испытательных методов

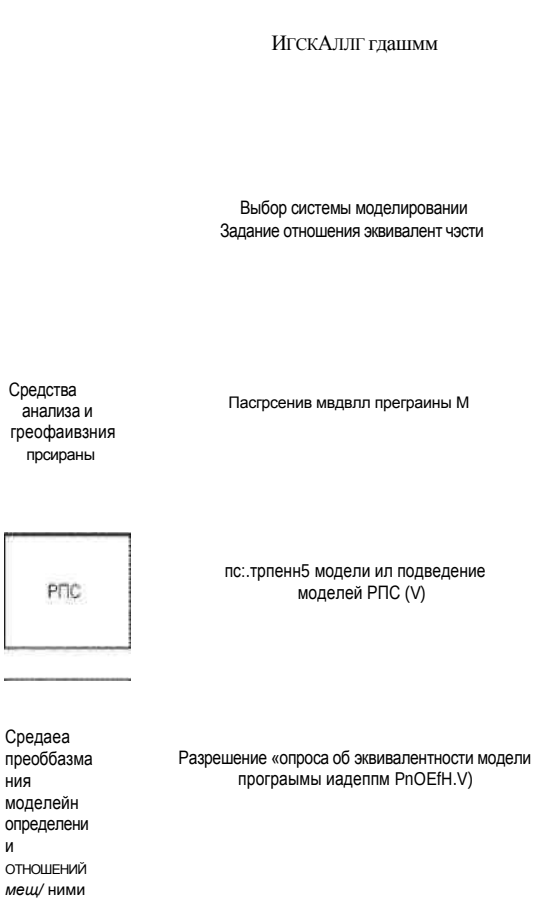

Рис. 23.4 - Схема анализа безопасности ПО с помощью логико-методов

*При проведении анализа безопасности с помощью **логико-аналитических методов** (см. рис.23.4) строится модель программы и формально доказывается эквивалентность модели исследуемой программы и модели РПС.*