4) Оператор связи совместно с органами контрразведки может производить мониторинг с целью обнаружения стеганографических каналов. В данном случае необходимо использовать весь арсенал средств противодействия стеганографическим атакам.

Заключение

Особенности организации стеганографических каналов в телефонных линиях связи определяются следующими условиями:

1) участком сети, на котором организуется скрытый канал:
   – участок абонентской линии;
   – коммутируемая телефонная сеть;
2) пространством возможных целей контролирующих систем:
   для абонента:
   – требование обеспечения качества связи;
   – защита от несанкционированного подключения;
   – защита своей информации;
   – защита от стеганографических каналов;
   для оператора связи:
   – обеспечение требуемого качества связи;
   – защита от несанкционированного доступа;
   – увеличение пропускной способности линии;
   – защита информации абонента;
   – защита от стеганографических каналов.

Анализ технических особенностей построения телефонных линий связи с учетом возможных целей контролирующих систем позволит создавать невидимые каналы передачи информации с минимальными затратами.

Список литературы

1. С.П. Расторгуев, Информационная война. – М.: Радио и связь, 1999.

2. В. О. Хорошко, О.Д. Азаров, М.Є. Шелест, Ю.Є. Яремчук, Основи комп'ютерної стеганографії. Навчальний посібник – Вінниця: ВДТУ, 2003.

3. Городская телефонная связь. Справочник // под. ред. А.С. Брискера и .П. Мельникова, М.: «Радио и связь», 1987.

УДК 004.621

А.С. Петров, О.А.Талыкин

АНАЛИЗ УЯЗВИМОСТЕЙ WEB-СИСТЕМ

В настоящее время Web-системы все чаще используются для доступа к информационным ресурсам предприятий. Динамическое внедрения Интернет технологий во все сферы деятельности человека ставит особые требования к обеспечению безопасности Web-систем.

Построение защищенных Web-систем требует проведения анализа нарушения безопасности и выявления причин возникновения уязвимостей. Только знание природы этих причин позволяет оценить способность Web-систем противостоять атакам на ее безопасность, а так же понять природу недостатков существующих средств обеспечения безопасности.

Целью исследования является анализ нарушений безопасности Web-систем, выявление причин возникновения уязвимостей и их систематизация.

Концептуальное определение Web-систем

Под Web-системой понимается распределенная информационная система, владеющая следующими свойствами:

- построение из компонентов, которые могут иметь разных владельцев;
- независимость одних компонентов от других;
- имеет способность использовать Web-системы и быть использованной другими Web-системами (быть компонентом иной системы);
- предполагает возможность появления в системе новых (прежде не определенных) пользователей, анонимных пользователей и доступ других систем как пользователей;
- физической и логической способностью функционировать как в глобальной, так и в локальной сетевой среде;
- базируется на использовании технологий гипертекста, мультимедиа, кроссплатформенных вычислений, распределенных объектных вычислений;
- является кроссплатформенной относительно использования аппаратных, сетевых и клиентских программных продуктов;
- является ориентированной на сетевые технологии относительно программных и информационных ресурсов.

Анализ существующих программных продуктов позволяет выделить как наиболее распространенные следующие три группы Web-систем:

1. **Статические Web-системы**. Данные системы предоставляют доступ на чтение пользователям к информационным ресурсам, но при этом не участвуют в процессе обработки информации, делегируя данные функции другим компонентам ВС.
2. **Прокси-системы**. Обеспечивают доступ пользователей к другим Web-системам, выступая посредником между пользователями и удаленными Web-системами с целью повышения производительности, уменьшения нагрузки на сеть и обеспечения безопасности.
3. **Динамические Web-системы**. Представляют собой полноценные системы обработки информации, участвуя в процессе обработки информации на всех этапах ее жизненного цикла.

Наименее не исследованной в области информационной безопасности и подверженной уязвимостям является динамическая Web-система.

В последнее время динамические системы все чаще вытесняют статические. Web-браузер перестал быть просто средством просмотра гипертекста, превратившись в универсальный инструмент обработки информации.

Архитектура построения динамических Web-систем носит индивидуальный характер и зависит от задач, стоящих перед системой, а также технологий, применяемых при ее реализации. В свою очередь можно выделить основные компоненты, присущие большинству существующих систем (табл. 1).

Компоненты динамической Web-системы Таблица 1

Web-система	ОС клиента	Web-браузер
		Другие приложения
	ОС сервера	Web-сервис
		Web-приложение
		Другие приложения и сервисы
	Сетевая среда	
	Информационные ресурсы	

С появлением приложений класса CMS (content management system) Web-приложения престали быть просто оболочками к информационным ресурсам, превратившись в самостоятельные системы обработки информации, реализующие практически все функции, присущие ВС. Но как показывает произведенный анализ существующих систем, в большинстве из них

требования к обеспечению безопасности не являются базовыми, а в некоторых просто отсутствуют.

В дальнейшем в качестве Web-системы будет рассматриваться Web-приложение, реализующее функции обработки информации, присущие вычислительным системам и являющейся Web-системой более низкого уровня абстракции.

Исследование уязвимостей

Под уязвимостью в дальнейшем будет пониматься совокупность причин, условий и обстоятельств, наличие которых в конечном итоге может привести к нарушению безопасности (несанкционированный доступ, ознакомление, уничтожение или искажение данных).

Исследованию ошибок, тем или иным образом связанных с безопасностью, всегда уделялось много внимания. В качестве примера можно привести работы М.Бишопа [6], Т.Аслама и И.Крсула [10], Зегжды Д.П. [1,2,3], Д. Ховарда [9]. Так в работе в рамках построения таксономии изъянов защиты (ИЗ) ВС К.Лендвером [11] была представлена классификация уязвимостей по источнику появления, приведенная в табл. 2.

К. Лендвер вместо термина «уязвимость» (vulnerability) использовал понятие «изъян защиты» (security flaw), определенный им как ошибка в программном продукте, позволяющая нарушителю обойти средства защиты.

Классификация источников появления ИЗ Таблица 2

<table>
<tr><td rowspan="8">Ошибки в системах защиты, служащие источником появления ИЗ</td><td rowspan="5">Преднамеренные</td><td rowspan="3">С наличием деструктивных функций (активные)</td><td rowspan="2">Разрушающие программные средства (РПС)</td><td>Несамовоспроизводящиеся РПС («троянские кони»)</td></tr>
<tr><td>Самовоспроизводящиеся РПС (вирусы)</td></tr>
<tr><td colspan="2">Черные ходы, люки, скрытые возможности проникновения в систему</td></tr>
<tr><td rowspan="2">Без деструктивных функций (пассивные)</td><td rowspan="2">Скрытые каналы утечки информации</td><td>С использованием памяти</td></tr>
<tr><td>С использованием времени</td></tr>
<tr><td colspan="3">Другие</td></tr>
<tr><td rowspan="5">Непреднамеренные (случайные)</td><td colspan="3">Ошибки контроля допустимых значений параметров</td></tr>
<tr><td colspan="3">Ошибки определения областей (доменов)</td></tr>
<tr><td colspan="3">Ошибки последовательности действий и использования нескольких имен для одного объекта (в том числе TOCTTOU)</td></tr>
<tr><td colspan="3">Ошибки идентификации/ аутентификации</td></tr>
<tr><td colspan="3">Ошибки проверки границ объектов</td></tr>
<tr><td colspan="4">Другие ошибки в логике функционирования</td></tr>
</table>

Под источником появления в данной работе понимается основа существования ИЗ, т.е. либо характеристики ВС, которые обуславливают его существование, либо принцип функционирования средств, использующих ИЗ для осуществления атаки.

В данной работе классификация преднамеренных ИЗ фактически представляет собой классификацию разрушающих программных средств по принципам функционирования, а непреднамеренных – классификацию ошибок, возникающих в ходе программной реализации.

Для решения практических задач и применительно к Web-системам наибольший интерес представляет классификация атак, предложенная в работе [12] созданной в рамках проекта webappsec.org.

Авторы данной работы ставили перед собой следующее цели:

- Определение всех известных классов атак на Web-приложения.
- Согласование названий для каждого из классов.
- Разработка структурированного подхода к классификации атак.

- Разработка документации, содержащей общее описание каждого из классов.

Данная классификация, представленная в табл. 3, содержит компиляцию известных классов атак, которые представляли угрозу для Web-приложений в прошлом и представляют сейчас. Каждому классу атак присвоено стандартное название и описаны его ключевые особенности. Классы организованы в иерархическую структуру.

Любая атака на вычислительную систему (подразумеваются успешные атаки, в результате которых происходит нарушение информационной безопасности) в том числе и на Web-систему опирается на определенные особенности построения и функционирования последней, иными словами — использует имеющиеся недостатки средств обеспечения безопасности.

Проведение анализа нарушений безопасности Web-систем позволяет при разработке и создании защищенных систем сконцентрировать основные усилия именно на устранение этих причин путем исправления в механизмах защиты выявленных недостатков, что позволяет эффективно противостоять угрозам безопасности. Очевидно, что основой данного подхода является глубокое исследование частных случаев нарушения безопасности и слабых сторон систем зашиты, сделавших возможным их осуществление.

Поскольку с точки зрения авторов причины успеха атак на системы обработки информации предопределены свойствами самих систем обработки информации, то анализ случаев нарушения безопасности должен основываться не столько на исследовании методов, используемых нарушителем, сколько на выявлении свойств системы, позволивших ему успешно осуществить атаку. Только знание природы этих свойств позволит оценить способность системы противостоять угрозам безопасности, а также понять недостатки в существующих средствах защиты, которые привели к соответствующим нарушениям, и построить защищенную систему, лишенную этих недостатков.

Классы атак на Web-приложения Таблица 3

Аутентификация Authentication)	Подбор (Brute Force)
	Недостаточная аутентификация (Insufficient Authentication)
	Небезопасное восстановление паролей (Weak Password Recovery Validation)
Авторизация (Authorization)	Предсказуемое значение идентификатора сессии (Credential/Session Prediction)
	Недостаточная авторизация (Insufficient Authorization)
	Отсутствие таймаута сессии (Insufficient Session Expiration)
	Фиксация сессии (Session Fixation)
Атаки на клиентов (Client-side Attacks)	Подмена содержимого (Content Spoofing)
	Межсайтовое выполнение сценариев Cross-site Scripting, XSS)
	Расщепление HTTP-запроса (HTTP Response Splitting)
Выполнение кода (Command Execution)	Переполнение буфера (Buffer Overflow)
	Атака на функции форматирования строк (Format String Attack)
	Внедрение операторов LDAP (LDAP Injection)
	Выполнение команд ОС (OS Commanding)
	Внедрение операторов SQL (SQL Injection)
	Внедрение серверных расширений (SSI Injection)
	Внедрение операторов XPath (XPath Injection)
Разглашение информа-	Индексирование директорий (Directory Indexing)

ции (Information Disclosure)	Идентификация приложений (Web Server/Application Fingerprinting)
	Утечка информации (Information Leakage)
	Обратный путь в директориях (Path Traversal)
	Предсказуемое расположение ресурсов (Predictable Resource Location)
Логические атаки (Logical Attacks)	Злоупотребление функциональными возможностями (Abuse of Functionality)
	Отказ в обслуживании (Denial of Service)
	Недостаточная проверка процесса (Insufficient Process Validation)

Целью создания таксономии причин возникновения уязвимостей Web-систем является ответ на вопрос: *что явилось причиной успешного осуществления нарушения безопасности в том или ином случае?*

Поэтому при проведении данного исследования случаев нарушений безопасности Web-систем ставились следующие задачи:

- определение этапов появления уязвимостей в Web-системе;
- определение компонентов Web-системы, в которых возникают и проявляются уязвимости;
- определение причин возникновения уязвимостей в Web-системе.

Только знание природы этих причин позволит оценить способность системы противостоять атакам на ее безопасность, а также понять природу недостатков, присущих существующим средствам обеспечения безопасности, которые привели к соответствующим нарушениям, и построить защищенную систему, лишенную этих недостатков.

В качестве базового множества случаев нарушений безопасности, которое послужило основой для построения таксономии, была использована статистика случаев нарушений безопасности, заимствованная из материалов CERT (Computer Emergency Response Team) [7].

Классификация уязвимостей по этапам внедрения

В отличие от исследования вопроса об источниках возникновения уязвимостей, анализу этапа появления ошибок уделяется недостаточное внимание. Результаты исследования данного вопроса подробно изложены в нескольких работах, наибольший интерес из которых представляет работа Лендвера [11], в которой проблема выявления этапа внедрения ошибок рассматривается по отношению к жизненному циклу программного обеспечения.

В работе Зегжды [1] предложена абстрактная схема, описывающая технологию разработки ПО, позволяющая выделить категории уязвимостей относительно времени их внедрения (табл. 4).

Классификация уязвимостей по этапу внедрения Таблица 4

Этапы внедрения ошибки и возникновения уязвимостей	На стадии разработки	Ошибки в требованиях и спецификациях
		Ошибки в исходных текстах программ
		Ошибки в исполняемом коде
	В ходе сопровождения	
	В ходе эксплуатации	

Так на самом верхнем уровне представления жизненного цикла систем выделяется три фазы:

1. Фаза разработки, которая охватывает весь период создания первой рабочей версии системы.

2. Фаза сопровождения, в ходе которой происходит модификация, совершенствование, развитие системы и появление ее очередной версии.

3. Фаза эксплуатации, то есть непосредственного функционального применения конкретной версии системы.

Требования к программному обеспечению описывают, что должна делать программа. Спецификации определяют то, каким образом эти действия должны выполняться.

На этапе составления требований и спецификаций чаще всего вносятся ошибки, обусловленные необходимостью реализации некоторых функциональных возможностей в ущерб безопасности системы.

Создание исходных текстов программ обеспечивает воплощение требований и спецификаций и является логическим продолжением соответствующих этапов. На данной стадии вносится большая часть уязвимостей, возникающих в результате неадекватной реализации или просто из-за ошибок программистов. Причем ошибки могут иметь как преднамеренный, так и случайный характер.

Генерация исполняемого кода в зависимости от языка программирования может осуществляться как в процессе разработки системы, так и в ходе эксплуатации (интерпретируемые языки). На данной стадии вносятся наиболее трудно выявляемые уязвимости, содержащиеся в самом компиляторе.

Возникновение уязвимостей в процессе сопровождения и развития системы. Данный этап наиболее уязвим для Web-систем, ввиду их особенности реализации. Добавление в систему новых функциональных возможностей, подключение дополнительных модулей (нередко и сторонних производителей) требует от разработчика тщательной проверки всей системы в целом.

Возникновение уязвимостей в процессе функционирования системы в большинстве случаев происходит по причине воздействия на нее разрушающих программных средств (РПС), пользователей ОС, на которой функционирует Web-система, или ошибок администрирования, что чаще всего приводит к получению злоумышленником доступа как к самой системе, так и к информационным ресурсам ею обслуживакмых.

Классификация уязвимостей по размещению в системе

Уязвимости можно классифицировать в зависимости от того, в каких компонентах системы они находятся.

Так как Web-система является распределенной и функционирует на нескольких ВС, то соответственно наследует и все уязвимости внешних к себе компонентов ВС, таких как ОС, аппаратное обеспечение, сетевая среда и т.д. В данной работе рассматриваются только те компоненты, которые могут содержаться непосредственно в динамической Web-системе (Web-приложении). Для этого был проведен анализ архитектур построения и выявлены следующие логические блоки, присущие большинству Web-систем (табл.5).

Классификация уязвимостей по размещению в Web-системе Таблица 5

Web- система	Уязвимости в средства защиты
	Уязвимости в базовых функциях, классах и компонентах
	Уязвимости в дополнительных модулях

Средства защиты. Набор функций, классов, модулей, алгоритмов и т.д. обеспечивающих безопасность обработки информации. Наличие уязвимостей в данном логическом блоке практически равносильно уязвимости всей системы.

Базовые функции, классы и компоненты. Под данным логическим блоком подразумеваются все базовые механизмы необходимые для функционирования системы и не имеющие отношения к безопасности системы.

Дополнительные модули. Наличие открытого и документированного интерфейса программирования, позволяющего реализовывать дополнительные функциональные возможности, является необходимым требованием для дальнейшего развития системы. В свою очередь подключение к базовому набору системы дополнительных модулей требует предварительного их изучения на уровне исходных текстов и тестирования работы всей системы в целом. Дополнительные модули могут содержать как ошибки программирования, так и разрушающие программные средства.

Таксономия причин возникновения уязвимостей

Проведенный анализ статистики случаев нарушений безопасности Web-систем показывает, что все случаи произошли по одной из следующих причин (табл.6):

1. **Выбор модели безопасности, не соответствующей назначению или архитектуре Web-системы**. Модель безопасности должна соответствовать как требованиям безопасности, так и принятой в ней технологии обработки информации.
2. **Неправильное внедрение модели безопасности**. Обычно неправильное внедрение модели безопасности в систему выражается в недостаточном ограничении доступа к наиболее важным для безопасности системы ресурсам, а также во введении различных исключений из предусмотренных моделью правил разграничения доступа.
3. **Отсутствие идентификации и/или аутентификации субъектов и объектов**. Во многих системах идентификация и аутентификация субъектов и объектов взаимодействия находится на весьма примитивном уровне – субъект может сравнительно легко выдать себя за другого субъекта и воспользоваться его полномочиями доступа к информации и механизмам управления системой. Часто идентификация и аутентификация носят непоследовательный характер и не распространяются на все уровни взаимодействия.
4. **Отсутствие механизмов локализации уязвимости в пределах компонента системы**. Наличие уязвимости в одном из компонентов системы автоматически приводит к уязвимости всей системы. Использование переменных, имеющих глобальную область видимости, позволяет компоненту системы, не связанному с осуществлением функций безопасности, вносить изменения в переменные содержащие такие данные, как: идентификатор пользователя, список групп, в которые входит данный пользователь, права доступа и т.д. Что дает возможность разработчикам модулей системы практически иметь полный контроль над всеми компонентам и информационными ресурсами.
5. **Ошибки, допущенные в ходе программной реализации средств обеспечения безопасности**. Эта группа причин нарушения безопасности будет существовать до тех пор, пока не появятся технологии программирования, гарантирующие производство безошибочных программ. Исчерпывающее тестирование и верификация программных продуктов позволяют сократить вероятность появления подобных ошибок.
6. **Наличие средств отладки и тестирования**. В связи со сложностью программных продуктов и практически невозможностью отладить все компоненты в лабораторных условиях многие программисты внедряют в обход существующих средств обеспечения безопасности механизмы, позволяющие проводить отладку и диагностику непосредственно в процессе эксплуатации. Очевидно, что для тех ситуаций, где безопасность имеет решающее значение, применение подобной практики является недопустимым.
7. **Отсутствие проверки допустимых значений параметров**. Многие классы атак, указанные в табл.3, осуществимы только потому, что во многих Web-системах отсутствует проверка полученных данных перед последующим их использованием.
8. **Ошибки администрирования**. Наличие самых совершенных средств защиты не предохраняет от возможных нарушений безопасности, т.к. в безопасности любой системы присутствует человеческий фактор: администратор, управляющий средствами обеспечения безопасности, может совершить ошибку, и все усилия разработчиков - будут сведены на нет. В свою очередь задача разработчиков предоставить интуитивно понятный и контролируемый механизм управления системой для уменьшения человеческого фактора.

Таксономия причин возникновения уязвимостей Таблица 6

Причины нарушения безопасности	Предопределенные на стадии разработки требований	Выбор модели безопасности, несоответствующей назначению или архитектуре
	Обусловленные принципами организации системы обеспечения безопасности	Неправильное внедрение модели безопасности
		Отсутствие идентификации и/или аутентификации субъектов и объектов
		Отсутствие механизмов локализации уязвимости в пределах компонента системы
	Обусловленные реализацией	Ошибки, допущенные в ходе программной реализации средств обеспечения безопасности
		Наличие средств отладки и тестирования
		Отсутствие проверки допустимых значений параметров
	Ошибки администрирования	

Предложенный подход к классификации причин нарушения безопасности в отличие от существующих подходов позволяет определить полное множество независимых первопричин нарушений безопасности, не сводимых одна к другой и образующих пространство факторов, определяющих реальную степень безопасности системы.

Из рис. 1 видно, что появление основных причин возникновения уязвимостей закладывается на этапе разработки, причем в основном на стадии задания спецификаций. Это вполне ожидаемый результат, т.к. именно этап составления спецификаций является одним из самых трудоемких, а последствия ошибок в спецификациях сказываются на всех последующих этапах разработки и распространяются на все взаимосвязанные компоненты системы. Так, отсутствие в системе механизмов локализации уязвимости сводит на нет все средства защиты в случае подключения модуля, имеющего потенциальные уязвимости. В то же время ошибки в исполняемом коде являются внешними для системы и наследуются системой от используемых компиляторов. Это требует использования безопасных версий компиляторов и их постоянного обновления в случае обнаружения уязвимостей.

Из рис. 2 видно, что ошибки администрирования и выбора модели безопасности играют обобщающий характер и воздействуют на всю систему целиком. В свою очередь отсутствие проверки допустимых параметров является наиболее часто встречающейся причиной нарушения безопасности.

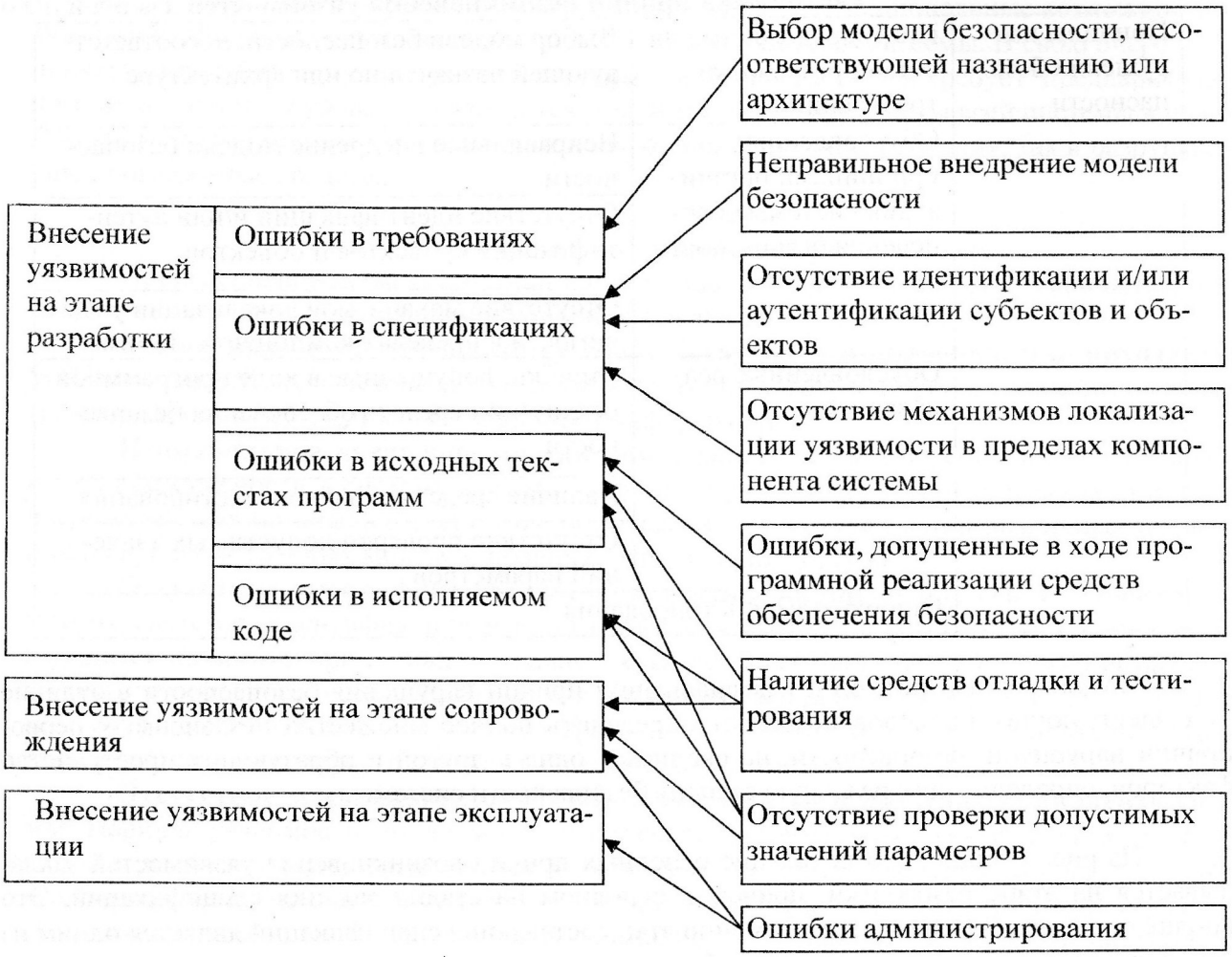

Рис.1. Взаимосвязь таксономии причин возникновения уязвимостей и классификации по этапам внедрения

Выводы

Представленная таксономия причин возникновения уязвимостей позволяет определить значимость каждой из причин и выявить этапы разработки Web-системы, на которых они могут быть устранены.

Установлено, что наибольшее значение имеют принципы организации защиты и управления доступом, т.е. те составляющие безопасности системы, которые закладываются на ранних стадиях определения требований, выбора модели безопасности и архитектуры средств защиты.

Разработанная таксономия причин возникновения уязвимостей является первым и необходимым этапом для решения задач построения защищенных Web-систем и может служить отправной точкой для адекватной реализации требований безопасности и корректной реализации средств защиты.

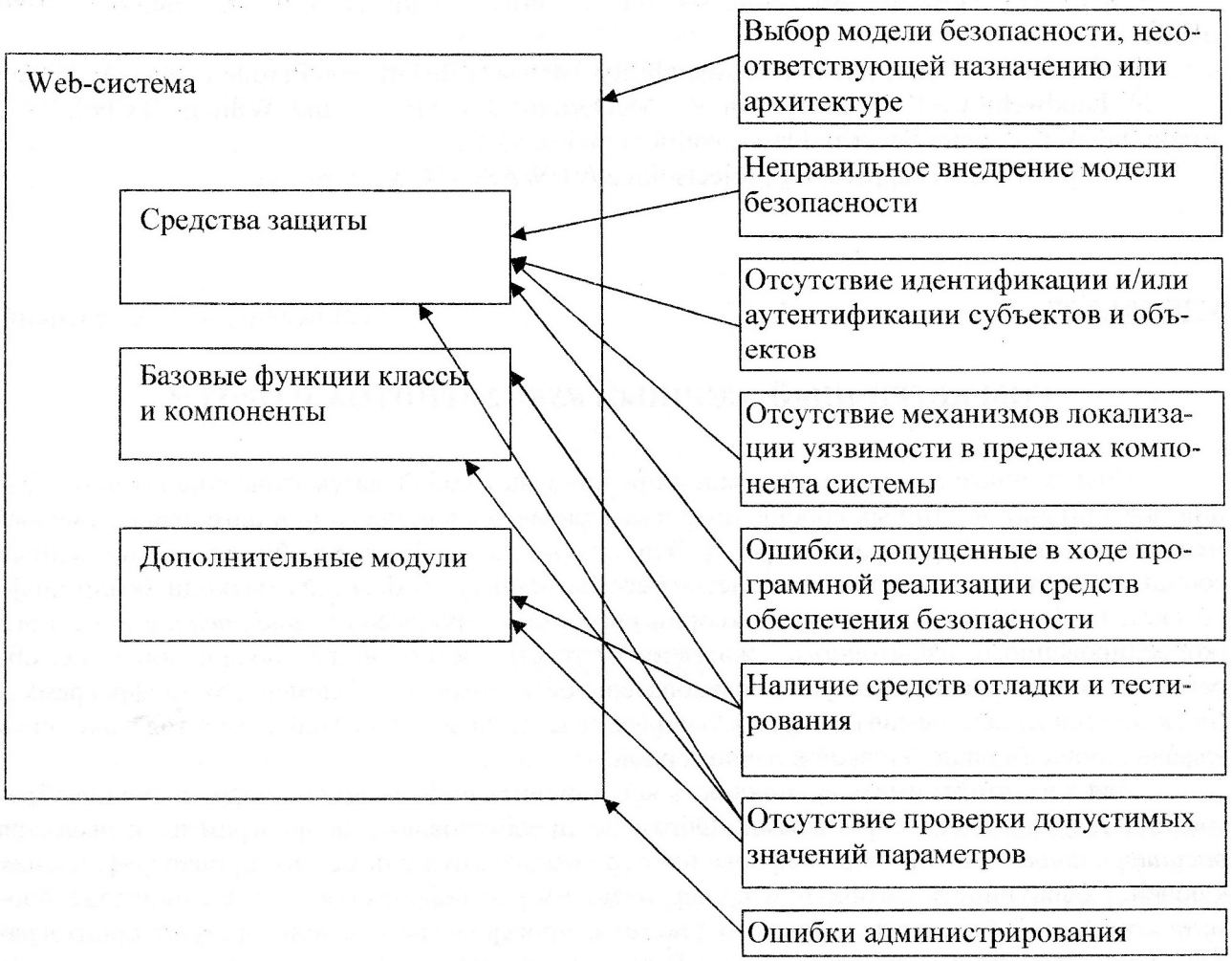

Рис.2. Взаимосвязь таксономии причин возникновения уязвимостей и классификации по размещению в системе

Список литературы

1.Зегжда Д.П., Ивашко А.М. Основы безопасности информационных систем // Горячая линия-Телеком, Москва, 2000 г. - 449 с.

2.Зегжда Д.П. и др. Под ред. Зегжды П. Д. Теория и практика обеспечения информационной безопасности // Издательство Агентства "Яхтсмен", М. 1996.-298с.

3.Зегжда Д.П. Общие принципы и теоретические основы анализа безопасности программ. В сборнике "Проблемы безопасности программного обеспечения" // Издание СПбГТУ, 1995 г., - С. 128-164.

4.Abbott R.P., Chin J.S., Donnelley J.E., Konigsford W.L., Tokubo S. and Webb D.A. 1976. Security analysis and enhancements of computer operating system. NBSIR 76-1041, National Bureau of Standards, ICST, April 1976.

5.Bisbey II, R. and Hollingworth, D. 1978. Protection analysis project report. ISI/RR-78-13, DTIC AD A056816, USC/Information Sciences Institute (May 1978).

6.Bishop Matt and Bailey Dave. A Critical Analysis of Vulnerability Taxonomies. Technical Report CSE-96-11, Department of Computer Science at the University of California at Davis, September 1996.

7.CERT, www.cert.org.

8.Handbook of INFOSEC Terms Version 2.0. CDROM, 1996. Stephen W. Hawking. A Brief History of Time: From the Big Bang to Black Holes. Bantam Books, 1988.

9.John D. Howard An Analysis of Security Incidents on the Internet 1989 -1995.

10. Krsul Ivan Victor. Software Vulnerability Analysis (PhD_thesis) Purdue University 1998.

11. Landwehr Carl E., Bull Alan R., McDermott John P. and William S.Choi. // A Taxonomy of Computer Security Flaws, with Examples. 93-94.

12. http://www.webappsec.org/projects/threat/v1/WASC-TC-v1_0.rus.txt

УДК 681.3.07

А.Я.Белецкий, А.А.Белецкий

СИММЕТРИЧНЫЙ БЛОЧНЫЙ *RSB*-32 КРИПТОАЛГОРИТМ

Современные методы шифрования представляют собой математические преобразования (алгоритмы), в которых сообщения рассматриваются как числа или алгебраические элементы в некотором пространстве [1, 2]. Эти алгоритмы отображают область «осмысленных сообщений» (исходный текст) в область «бессмысленных сообщений» (выходной или шифротекст, шифрограмма). С позиций теории сигналов и процессов *зашифрование* исходного (коррелированного, избыточного, сжимаемого) текста состоит в его «обеливании», т.е. обращении в некоррелированную последовательность символов (элементов) шифрограммы (практически несжимаемой) с плотностью распределения вероятностей элементов выходного алфавита максимально близкой к равномерной.

Для того чтобы иметь возможность восстановить информацию, шифрующие преобразования должны быть обратимыми. Обратное преобразование шифрограммы называется *расшифрованием*. Алгоритмы шифрования параметризуются с помощью криптографических ключей. Совокупность алгоритмов зашифрования и расшифрования, а также описание формата сообщений (входного открытого текста) и пространства ключей образуют криптографическую систему, или *криптосистему*. В том случае, когда для зашифрования и расшифрования используется один и тот же ключ (или ключ расшифрования достаточно легко вычисляется из ключа зашифрования), то такие криптосистемы называются *криптосистемами с секретным* (симметричным) *ключом*.

В данной статье предлагается достаточно гибкая к изменению параметров шифрования (размеров ключей, блоков и элементов замены в блоках) симметричная блочная криптосистема, названная системой ***RSB*-32**. Аббревиатура ***RSB*** происходит от ключевых слов ***Round, Step, Bloc*** – подчеркивая тем самым, что основными для криптоалгоритма являются раундовые преобразования ($\boldsymbol{R}$), разбитые на определенное число шагов ($\boldsymbol{S}$), а действие алгоритма осуществляется над блоками ($\boldsymbol{B}$) открытого или закрытого текстов, причем размер раундового ключа (как элемента общего ключа) составляет 32 бита.

Общее описание *RSB* криптосистемы. ***RSB*** – это итерационный блочный шифр, который доставляет уникальную возможностью по изменению как размеров блоков и ключей, так и размеров элементов блоков, над которыми (элементами) выполняются нелинейные операции замен. В отличие от большинства известных симметричных шифров в ***RSB*** криптосистеме таблицы (матрицы) замен остаются не постоянными, а изменяются в зависимости от состояния секретного ключа.

Основные параметры ***RSB*** шифра:

- Длина раундового ключа - 32 бита.
- Длина общего (шагового) ключа: $r*64$, $r = 1, 2, ...$
- Число шагов шифрования: $s = 1, 2, ...$
- Число раундов шифрования: $r*s$.
- Размер блока: 256, 512, 1024 бита.
- Размер элементов замены: 8, 16, или 32 бита, т.е. ***RSB*** криптосистема