УДК 004.056
DOI 10.53403/9785951505200_584

# АНАЛИЗ УЯЗВИМОСТЕЙ КАК НЕОБХОДИМЫЙ ЭТАП РАЗРАБОТКИ СИСТЕМЫ ПОЛНОГО ЖИЗНЕННОГО ЦИКЛА ИЗДЕЛИЙ

*Кротов Владислав Игоревич (VIKrotov@vniief.ru)*

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

В данной работе рассмотрен анализ уязвимостей как один из основных этапов разработки системы полного жизненного цикла изделий. Описываются особенности анализа уязвимостей программного обеспечения (ПО) и информационной системы (ИС), приводятся классификации уязвимостей и методы анализа уязвимостей. Проведен анализ процесса организации системы защиты, показана взаимосвязь между уязвимостью и угрозой ИБ, к которой может привести данная уязвимость.

**Ключевые слова:** информационная система, информационная безопасность, анализ уязвимостей, угроза, эксплуатация уязвимостей, уязвимость, безопасное программное обеспечение, защита информации, система защиты информации, атака.

# VULNERABILITY ANALYSIS AS A NECESSARY STAGE IN DEVELOPING A FULL LIFECYCLE SYSTEM OF PRODUCTS

*Krotov Vladislav Igorevich (VIKrotov@vniief.ru)*

FSUE “RFNC-VNIIEF”, Sarov Nizhny Novgorod region

This paper considers the analysis of vulnerabilities as one of the main stages in the development of a full lifecycle system of products. The features of the analysis of software and information system vulnerabilities are described, the classification of vulnerabilities and methods of vulnerability analysis are given. The analysis of the process of organizing the protection system is carried out, the relationship between the vulnerability and the IS threat to which this vulnerability can lead, is shown.

**Key words:** information system, information security, vulnerability analysis, threat, exploitation of vulnerabilities, vulnerability, secure software, information security, information security system, attack.

## Введение

Современное развитие в области информационной безопасности (ИБ) сопровождается постоянным ростом и совершенствованием компьютерных атак, которые приводят к снижению защищенности ресурсов автоматизированных систем. Главной причиной успешных компьютерных атак является наличие уязвимостей в разрабатываемом программном обеспечении (ПО). Небезопасная программа позволяет злоумышленникам использовать имеющиеся уязвимости для просмотра, изменения или удаления информации, а также внедрять свой код в систему. Применение принципов безопасной разработки позволяет защититься от этого.

Вопросы обеспечения безопасности и работоспособности системы являются неотъемлемой частью этапа ее проектирования. Поэтому проблема разработки безопасного ПО является одной из основных в области ИБ, требующей решения целого ряда задач. Следовательно, обеспечение защищенности информационных систем (ИС) является одним из важнейших этапов ее разработки и поддержки, а анализ уязвимостей в ходе всего жизненного цикла позволяет обнаруживать и устранять потенциальные слабости программы, которые могут быть использованы для реализации угроз безопасности информации.

## Безопасное ПО

На этапе проектирования системы защиты информации важно не упустить существенных деталей и, в то же время, не переоценить некоторые из них. Необходимо знать о характере возможных опасностей, классифицировать угрозы и проводить меры по их устранению.

Для защиты информации от угроз и обеспечения необходимого уровня безопасности информации требуется реализация мер, направленных на предотвращение появления и устранение уязвимостей программ в процессах жизненного цикла ПО [1].

Следует отметить, что для разработки ПО применяются разнообразные инструментальные средства, также имеющие слабости, которые могут привести к реализации угроз для ПО и ИС.

Безопасное ПО – ПО, которое разработано с использованием мер, направленных на предотвращение появления и устранения уязвимостей программы (систематический поиск уязвимостей программы, исправление обнаруженных уязвимостей программы) [2].

Для организации работ, выполняемых в процессах жизненного цикла ПО и подтверждения соответствия требованиям стандарта документация разработчика ПО должна содержать перечень определенных требований по безопасности, предъявляемых к разрабатываемому ПО.

В качестве источников для формирования требований разработчик ПО может использовать требования законов, нормативных правовых актов, отраслевых стандартов, перечень требований пользователя, сценарии применения ПО. Например, могут быть определены требования по безопасности:

– к обеспечению идентификации и аутентификации;

– к обеспечению защиты от несанкционированного доступа к информации;

– к обеспечению регистрации событий;

– к контролю точности, полноты и правильности данных, поступающих в программу;

– к обработке программных ошибок и исключительных ситуаций.

Для проверки ПО на соответствие сформированным требованиям выполняются исследования по выявлению уязвимостей и недекларированных возможностей.

## Система защиты информации, особенности анализа уязвимостей

Для обеспечения нейтрализации угроз безопасности информации с использованием требуемых методов и способов защиты, используется система защиты информации (СЗИ). СЗИ – это комплекс организационных и технических мер, направленных на обеспечение информационной безопасности предприятия. Главным объектом защиты являются данные, которые обрабатываются в автоматизированной системе управления (АСУ) и задействованы при выполнении бизнес-процессов.

Основными мероприятиями по защите информации [3] при ее обработке в ИС являются:

– определение вида информации, обрабатываемой в ИС;

– определение угроз безопасности информации и разработка частной модели угроз;

– разработка СЗИ в соответствии с требованиями по защите информации (ЗИ);

– разработка эксплуатационной, технической и организационно-распорядительной документации по СЗИ;

– внедрение СЗИ;

– обучение лиц, эксплуатирующих средства защиты;

– аттестация ИС по требованиям безопасности информации;

– ввод СЗИ в эксплуатацию;

– контроль за соблюдением требований по ЗИ и условий использования средств защиты;

– составление заключений по фактам нарушений, приводящим к снижению уровня защищенности информации, разработка и принятие мер по предотвращению подобных нарушений.

Грамотно организованная и высокоэффективная СЗИ способна обеспечить защиту от кражи любых видов информации из корпоративной, внутренней сети компании. Организации, которые заинтересованы в получении эффективной СЗИ, постоянно работают над тем, чтобы предотвратить:

– утечки различных видов защищаемой информации;

– удаленные изменения защищаемых данных;

– изменения уровня защиты от киберугроз.

Анализ уязвимостей – процесс, направленный на обнаружение всевозможных угроз, уязвимых мест и рисков вероятного несанкционированного проникновения третьих лиц в ИС. Уязвимость выступает в качестве слабого места ИС. Угроза – фактор оказания отрицательного воздействия, которое потенциально становится причиной компрометации конфиденциальных и других видов защищаемых данных.

Анализ уязвимостей является неотъемлемой частью СЗИ и позволяет выявить недостатки и снизить угрозы безопасности.

## Уязвимости, методы анализа уязвимостей

Под уязвимостью понимают недостаток (слабость) программного (программно-технического) средства или ИС в целом, который (которая) может быть использована для реализации угроз безопасности информации [4, 6].

Не устраненные уязвимости для киберпреступников – настоящая находка. Воспользовавшись брешами в защите (например, из-за неустановленного вовремя патча), злоумышленник может проникнуть в информационно-технологическую (ИТ) инфраструктуру компании, а дальнейшие шаги могут быть самыми разными.

В основе классификации уязвимостей ИС используются следующие классификационные признаки: область происхождения, типы недостатков ИС и место возникновения уязвимости ИС [5]. К основным поисковым признакам уязвимостей ИС относятся: наименование операционной системы и тип аппа-

ратной платформы, наименование ПО и его версия, степень опасности уязвимости.

Уязвимости ИС по области происхождения подразделяются на следующие классы:

– уязвимости конфигурации (уязвимость, появившаяся в процессе задания конфигурации (применения параметров настройки) ПО и технических средств ИС);

– архитектурные уязвимости (уязвимость, появившаяся в процессе проектирования ИС);

– уязвимости кода (уязвимость, появившаяся в процессе разработки ПО).

– организационные уязвимости (уязвимость, появившаяся в связи с отсутствием (или недостатками) организационных мер ЗИ в ИС и (или) несоблюдением правил эксплуатации системы защиты информации ИС, требований организационно-распорядительных документов по защите информации и (или) несвоевременном выполнении соответствующих действий должностным лицом (работником) или подразделением, ответственным за защиту информации);

– многофакторные уязвимости (уязвимость, появившаяся в результате наличия нескольких недостатков различных типов).

Существует общий перечень дефектов безопасности (CWE – Common Weakness Enumeration), который представляет собой официальный реестр или словарь общих дефектов безопасности, которые могут проявиться в архитектуре, проектировании, коде или реализации ПО, и могут быть использованы злоумышленниками для получения доступа к ИС. Данный перечень выступает в качестве универсального формального языка для описания дефектов безопасности ПО, а также для распознавания, устранения и предотвращения этих дефектов.

Помимо перечня дефектов безопасности CWE существуют также официально признанные реестры и базы данных (БД) уязвимостей: NDV, CVE, VulnDB, X-Force, Банк данных угроз безопасности информации ФСТЭК России и другие. Широкое применение получили различные системы классификации и оценки критичности уязвимостей: NIPC, SANC, nCircle, CVSS, WIVSS и другие.

Исследования по выявлению уязвимостей, предусматривающие контроль их отсутствия, в общем случае, следующие:

• Экспертный анализ (моделирование атак, направленный анализ участков кода, анализ документации) – это анализ информационных источников, содержащих сведения об уязвимостях ПО, анализ доступных исходных данных. Результатом экспертного анализа является набор потенциальных уязвимостей приложения.

• Статический анализ (межпроцедурный контекстно-чувствительный анализ, синтаксический анализ, формальная верификация, анализ бинарного кода, анализ байт-кода, анализ, чувствительный к путям выполнения) – это анализ программного обеспечения, производимый (в отличие от динамического анализа) без реального выполнения исследуемых программ. В большинстве случаев анализ производится над какой-либо версией исходного кода. В зависимости от используемого инструмента глубина анализа может варьироваться от определения поведения отдельных операторов до анализа, включающего весь имеющийся исходный код. Способы использования полученной в ходе анализа информации также различны – от выявления мест, возможно содержащих ошибки (утилиты типа Lint), до формальных методов, позволяющих математически доказать какие-либо свойства программы (например, соответствие поведения спецификации). Результатом статического анализа могут быть пути выполнения ПО (которые потом используются в динамическом анализе), графы и матрицы (анализируются вручную), набор потенциальных уязвимостей и перечень актуальных уязвимостей.

• Динамический анализ (фаззинг, отслеживание помеченных данных, сканирование уязвимостей, анализ активности и потоков взаимодействия программы, эмуляция оборудования) – это анализ программного обеспечения, производящийся при помощи выполнения программ на реальном или виртуальном процессоре (в отличие от статического анализа). Утилиты динамического анализа могут требовать загрузки специальных библиотек, перекомпиляцию программного кода. Некоторые утилиты могут инструментировать исполняемый код в процессе исполнения или перед ним. Для большей эффективности динамического анализа требуется подача тестируемой программе достаточного количества входных данных, чтобы получить более полное покрытие кода. Результатом динамического анализа является набор потенциальных уязвимостей и перечень актуальных уязвимостей.

• Антивирусное сканирование – это метод заключается в использовании средств антивирусной защиты с целью выявления вредоносных компьютерных программ (вирусов) в ПО. Применяется для отслеживания активности всех системных и пользовательских процессов в момент установки ПО, а также в сигнатурном и эвристическом анализе содержимого оперативной памяти, файловой системы и загрузочных секторов всех носителей по завершении установки ПО.

• Сопоставление статических и динамических маршрутов.

Рассмотрим подробнее методы поиска недостатков приложения доступные внешнему пользователю (злоумышленнику). Это методы, анализирующие работу приложения без обращения к исходным кодам, среди них самыми популярными и доступными являются:

– метод получения идентифицирующей информации о приложении и выявление его слабостей с помощью общедоступных реестров и баз данных уязвимостей;

– сканирование ИС анализаторами уязвимостей с открытым исходным кодом;

– метод тестирования на проникновение.

### Получение информации о приложении и выявление его слабостей с помощью общедоступных реестров и баз данных уязвимостей

Данный метод ввиду своей простоты и доступности получил наиболее широкое применение при проведении атак. Под атакой понимается любое действие нарушителя, которое приводит к реализации угрозы путём использования уязвимостей ИС.

Метод основан на посылке от имени пользователя запросов приложению, ответы на которые позволяют получить информацию о сервере, о том на какой технологии разработано приложение, какие версии ПО использует, какие компоненты и др.

Возможность получения идентифицирующей информации пользователем приложения является потенциальной уязвимостью, как говорилось выше, в сети Интернет накоплены огромные объёмы данных по уязвимостям программных продуктов с указанием версий программ (те же самые официальные реестры и БД уязвимостей), методов реализации атак (эксплойтов), кроме того ежедневно публикуются отчёты о новых найденных уязвимостях. Эта информация в основном предназначена для администраторов и специалистов в области информационной безопасности, в тоже время она доступна через сеть Интернет всем желающим. Администраторы информационных систем часто не устанавливают обновления безопасности вовремя. В результате становится возможным обойти защиту информационной системы путем использования всем известной уязвимости, которую администратор не закрыл установкой соответствующего обновления.

### Сканирование приложения анализаторами уязвимостей

Метод заключается в применении сканеров уязвимостей, результатом работы которых является выявление потенциальных уязвимостей ПО. Сканер, как правило, на основе имеющихся в его составе базы данных признаков получает идентификационные данные приложений (версии программ и библиотек, входящих в состав ПО, а также особенности конфигурации). На основе данной информации и сведений, которые содержаться в БД известных уязвимостей приложений, определяется наличие потенциальных уязвимостей. Примерами таких сканеров являются: APT2, sqlmap, nmap, OpenVAS.

### Тестирование на проникновение

Тестирование на проникновение (тесты на преодоление защиты, penetration testing, pentest, пентест) – метод оценки безопасности компьютерных систем или сетей средствами моделирования атаки злоумышленника [8]. Суть таких работ заключается в санкционированной попытке обойти существующий комплекс средств защиты информационной системы.

Процесс включает в себя активный анализ системы на наличие потенциальных уязвимостей, которые могут спровоцировать некорректную работу целевой системы, либо полный отказ в обслуживании. При этом исследуются все компоненты системы, а также взаимосвязи между ними (особенно на границах доверенной зоны). В ходе тестирования на проникновение разрабатываются сценарии типовых атак, которые могут быть выбраны из перечня типовых атак (например из базы Common Attack Pattern Enumeration).

Тестирование на проникновение можно разделить на две группы:

1. Инфраструктура (межсетевые экраны, маршрутизаторы, системы, веб-серверы, базы данных, серверы приложений, любой компонент или устройство, от которого зависит сервис в целом);

2. Приложение (компоненты на стороне клиента).

При поиске уязвимостей методом тестирования на проникновение циклически выполняются следующие стадии: планирование, сбор информации (идентификация уязвимостей), проведение атак на систему.

Инструменты для тестирования на проникновение также находятся в открытом доступе, например, уже в предустановленном виде и готовые к использованию они имеются в операционной системе Kali Linux.

### Эксплуатация уязвимостей

Наличие уязвимости в ПО в ИС, может позволить обойти средства защиты СЗИ. Эксплуатация уязвимостей – это способ взять компьютер под контроль, даже если запущенное в данный момент приложение способно предотвращать подобные вещи. Программы умеют делать только то, для чего они были спроектированы, соответственно, недостатки в безопасности – это слабые места или недочеты в конструкции самой программы или той среды, в которой она выполняется.

В качестве примера эксплуатации уязвимости, рассмотрим уязвимость смещения на единицу (off-by-one error) в оболочке OpenSSH, которая задумывалась как набор программ для защищенной связи с терминалом, предназначенный для замены таких небезопасных и нешифрованных служб, как telnet, rsh и rc. Код оператора if выглядел следующим образом:

*if (id < 0 || id > channels_alloc) {*

На самом деле требовалось вот такое условие:

*if (id < 0 || id >= channels_alloc) {*

На обычном языке этот код означает: «*Если идентификатор меньше 0 или больше числа выделенных каналов, делайте следующее…*». А нужно было написать: *«Если идентификатор меньше 0 или больше числа выделенных каналов либо равен ему, делайте следующее…*».

Эта ошибка позволила обычным пользователям получать в системе права администратора. Разумеется, разработчики защищенной программы OpenSSH не собирались добавлять в нее такой возможности, но компьютер делает только то, что ему приказано [7].

Эксплуатация уязвимостей в операционных системах и приложениях является одним из самых популярных методов киберпреступников на сегодняшний день. С целью повышения вероятности заражения создатели вирусов разрабатывают и продают наборы эксплойтов, которые уже нацелены сразу на несколько слабых мест в системе. Со временем новые эксплойты добавляются в уже существующие и распространенные наборы, что позволяет легко обходить защиту, используя недостатки в ИС.

## Заключение

В данной работе проведен анализ уязвимостей ИС, описаны их особенности, проанализированы методы выявления уязвимостей. Стоит отметить, что методы поиска недостатков приложения доступные внешнему пользователю (злоумышленнику) позволяют находить и эксплуатировать уязвимости. Была показана взаимосвязь между уязвимостью ИС и угрозой информационной безопасности, а именно, как можно эксплуатировать уязвимости приложений, если вовремя не устанавливать обновления используемых сервисов, программ ИС (не соблюдать требования СЗИ), не проводить анализ уязвимостей для выявления слабостей и недостатков ИС.

На основании проведенного анализа можно сделать вывод, что анализ уязвимостей ИС является важным этапом жизненного цикла изделия.

## Список литературы


1. ГОСТ Р 56939-2016. Защита информации. Разработка безопасного программного обеспечения. Общие требования.
2. ГОСТ Р ИСО/МЭК 18045. Информационная технология. Методы и средства обеспечения безопасности. Методология оценки безопасности информационных технологий.
3. ГОСТ Р 56546-2015.Уязвимости информационных систем. Классификация уязвимостей информационных систем.
4. ГОСТ Р 56545-2015. Уязвимости информационных систем. Правила описания уязвимостей.
5. ГОСТ Р ИСО/МЭК 18045-2013. Информационная технология. Методы и средства обеспечения безопасности информационных технологий.
6. ГОСТ Р 58142-2018. Информационная Технология. Методы и средства обеспечения безопасности. Детализация анализа уязвимостей программного.
7. ГОСТ Р 58142-2018, часть 1. Использование доступных источников для идентификации потенциальных уязвимостей.
8. ГОСТ Р 58143-2018, часть 2. Тестирование проникновения.