ТЕМА 2: ВВЕДЕНИЕ В БЕЗОПАСНОСТЬ ВЕБ-ПРИЛОЖЕНИЙ

Autor: N.Plești, lector univ.

Содержание

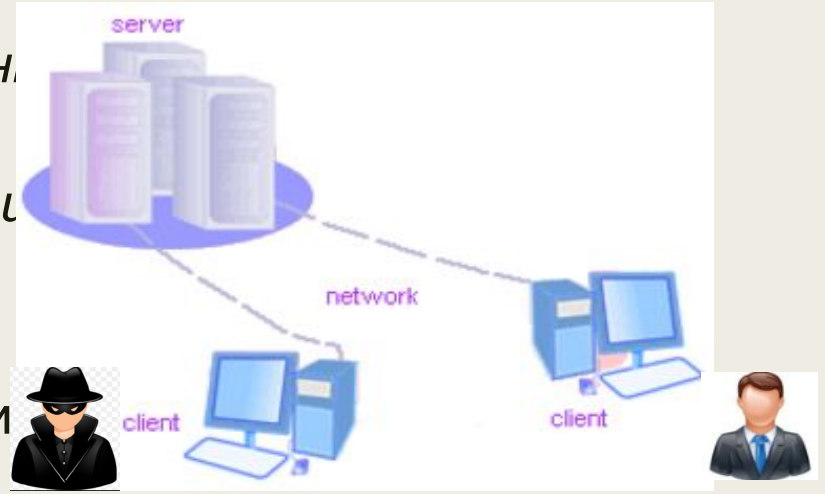
- Определение понятий.
- Риски, связанные с использованием веб-приложений.
- Общие проблемы, с которыми сталкиваются организации в процессе обеспечения безопасности приложений, доступные через веб-интерфейсы.
- Рекомендуемые принципы которые необходимо соблюдать в процессе разработки безопасных веб приложений.

Организации и веб

- **Веб-приложени**я представляют собой динамические веб-сайты, которые содержат элементы программирования на стороне сервера и обеспечивают функциональные возможности для взаимодействия с пользователями, подключения к базам данных и создания веб-контента (результатов), чаще всего динамически для браузеров (клиентов)
- Сегодня деятельность любой организации не может быть представлена без использования веб-приложений, реализованных для различных бизнес-процессов: презентационные сайты, электронные магазины, банковские системы, интернет-банкинг, продажи и т. д.
- Много корпоративных приложений или компьютерных информационных систем, которые до недавнего времени использовались в качестве настольных (desktop) приложений, в последнее время, путем модернизации, стали использовать веб-технологии
- Дополнительно https://habr.com/company/haulmont/blog/354992/

Необходимость создания безопасных веб приложений

- Процент использования в организациях веб-приложений каждый год растет, но есть один важный аспект использования веб-приложений:
  - *несанкционированный доступ к приложению может привести к потере доверия к организации или*
  - *организация может потерять часть своих клиентов (из-за неправильного хранения их личн данных) или*
  - *организация может иметь несколько прямых или косвенных финансовых потерь (для восстановления вандализированных ресурсов)*
- По этим причинам разработка безопасного приложени имеет большое значение, наряду с целью разработки функционального веб-приложения



Безопасность веб приложений

- **Безопасность веб-приложений** - это подразделение информационной безопасности, которая занимается, в частности безопасностью веб-сайтов, веб-приложений и веб-сервисов
- На высоком уровне безопасность веб-приложений основана на принципах безопасности приложений, но эти принципы специально применяются к информационным системам и приложениям, доступным пользователям через веб-интерфейсы
- В начале 2000-х годов, с появлением Web 2.0, который имеет в качестве базовой функции участие пользователя в создании и / или изменении веб-контента (например, социальные сети), приводит к увеличению обмена информацией через компьютерные сети
- Это значительно увеличивает количество услуг и бизнеса, управляемых через Интернет. По этой причине хакеры все чаще пытаются скомпрометировать цифровые ресурсы и корпоративные сети

Определение понятия

- **Безопасность приложений** включает меры, направленные на **повышение безопасности приложения, путем обнаружения, удаления или предотвращения уязвимостей системы безопасности**
  - *Для преодоления таких уязвимостей безопасности, используются разные технологии на разных этапах жизненного цикла приложений*
- **Уязвимость** является недостатком приложения, которое возникает в результате дефекта проектирования или ошибки реализации
- Уязвимость позволяет злоумышленнику нанести вред заинтересованным сторонам в процесе использовании приложения. Заинтересованные стороны включают владельца приложения, пользователей приложения и других участников, которые используют приложение
- Термин «уязвимость» используется очень редко. Тем не менее, такие понятия как «угрозы», «атаки» и «контрмеры» могут быть включены в процесс борьбы с ними

Примеры уязвимостей

- Отсутствие проверки входных данных пользователя
- Отсутствие достаточного механизма аутентификации и регистрации действий пользователей в процесе использования приложений
- Неверная обработка ошибок
- Соединение с базой данных не закрывается должным образом
- По итогам 2015 года среди абсолютных лидеров можно отметить уязвимости межсайтового выполнения сценариев (в результате таких атак злоумышленник может, например, получить доступ к личному кабинету пользователя и выполнять мошеннические операции), что обнаружено в 80% проанализированных специалистами приложений. Или же разглашение чувствительной информации, риск которого присутствует в 50% приложений. А также атаки на включение внешних сущностей в XML, характерные для 40% изученных приложений. Последнее, кстати, позволяет реализовать широкий спектр угроз вплоть до получения полного контроля над атакуемым сервером. В общей сложности более 70% всех обследованных приложений содержали уязвимости высокой степени риска (в России)
- Больше информаций об уязвимостях https://www.owasp.org/index.php/Top_10-2017_Top_10

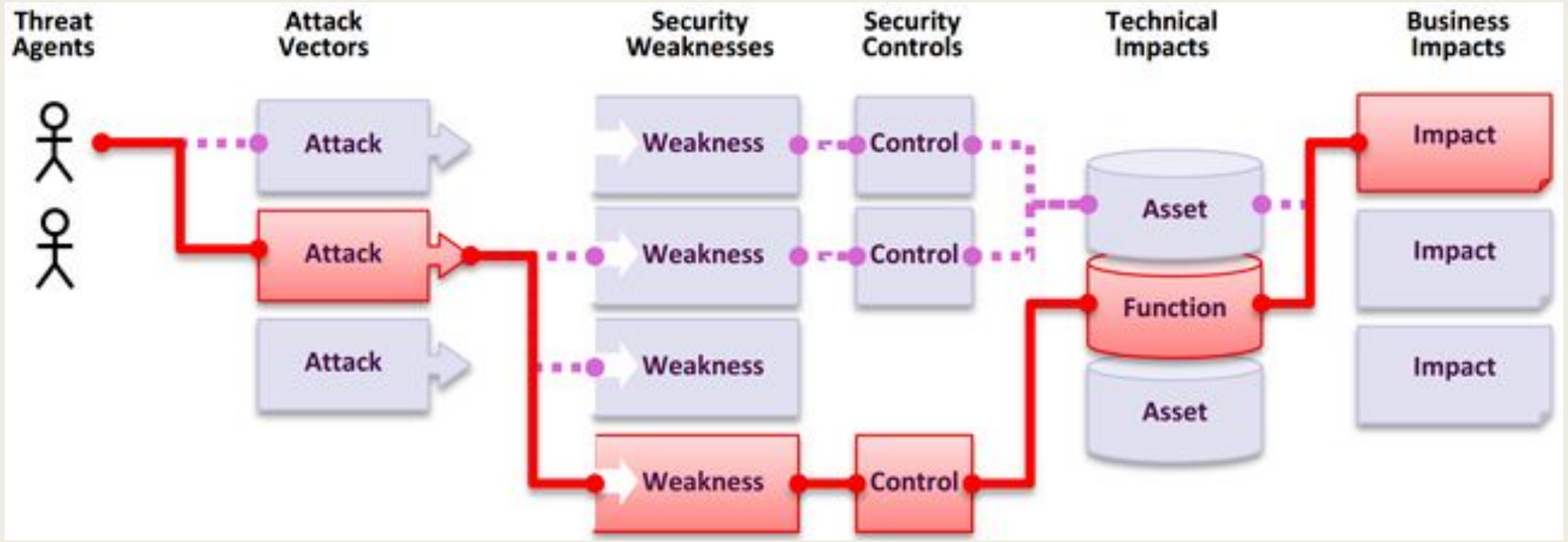
OPEN WEB APPLICATION SECURITY PROJECT

- Среди разработчиков появились рекомендации, к обеспечению безопасности WEB-приложений, которые вылились в проект под названием: Open Web Application Security Project (OWASP)
- OWASP – это открытый проект обеспечения безопасности WEB-приложений, который включает в себя корпорации, образовательные организации и индивидуальных разработчиков, которые совместными усилиями формируют статьи, рекомендации и учебные пособия, находящиеся в свободном доступе и рекомендуемы при разработке WEB-приложений. Также проект включает в себя ряд тренировочных задач и инструменты для анализа безопасности WEB-приложений

Риски которые угрожают веб приложениям

- Атакующие могут использовать несколько способов проникновения в веб-приложение, чтобы нанести вред бизнесу или организации
- Каждый из этих путей представляет собой **риск**, который может быть достаточно серьезным, чтобы привлечь внимание разработчиков веб-приложений
- Иногда эти пути легко найти и использовать, а иногда - сложнее. Аналогичным образом, ущерб, причиненный этими рисками, может не иметь серьезных последствий или в других случаях может привести к разрушению бизнеса
- Чтобы определить риск для организации, можно оценить вероятность проявления риска, которая зависит от исходного элемента угрозы, вектора атаки и уязвимости безопасности, в сочетании с оценкой технических потерь и воздействий на бизнес организации. Все эти факторы вместе определяют общий риск

Факторы, влияющие на риск



Уязвимости vs риски

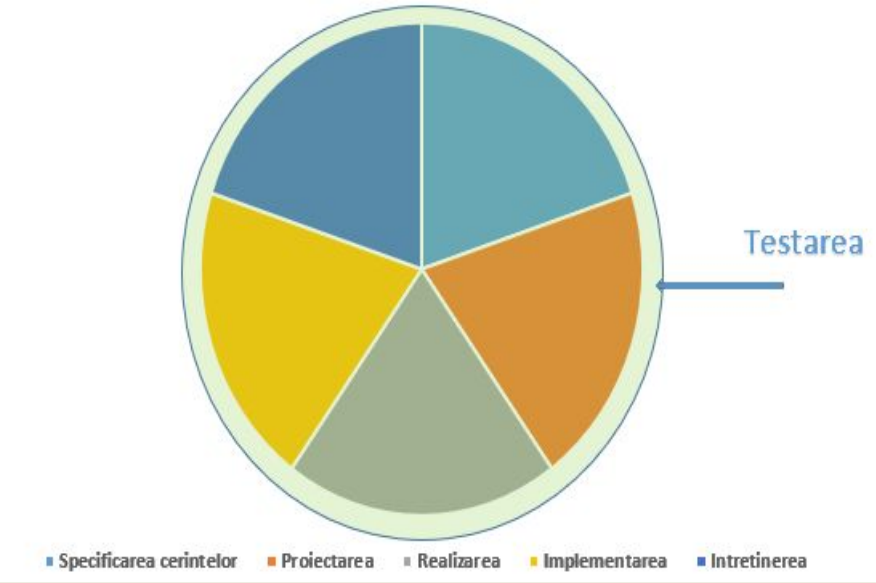
- Важное значение имеет обнаружение уязвимостей. Также необходима оценка влияния риска на бизнес
- В начале жизненного цикла можно определить проблемы, связанные с безопасностью архитектуры или проектированием решения, используя «моделирование угроз». Позже проблемы безопасности можно найти, используя проверку кода или тестирование на проникновение. Или проблемы могут и не быть обнаружены до тех пор, пока приложение не будет выпущено в производство, а затем оно может быть скомпрометировано
- Риск можно оценить, используя формулу :

Риск = ВероятностьПоявления * Воздействие

  - *OWASP анализирует значения вероятности от 0 до 9*

Жизненный цикл защищенного веб-приложения

- Большинство разработчиков веб-ресурсов не следуют рекомендациям по обеспечению безопасности приложений на каждом этапе её жизненного цикла и в результате они получают и используют веб-приложения, которые содержат уязвимости, которые можно избежать на ранних стадиях жизненного цикла. Наличие этих уязвимостей - это вредоносные «двери» для доступа злоумышлинников к данным приложения - иногда к частным или другим информационным ресурсам
- Поэтому тот же проект OWASP рекомендует оценивать безопасность приложений на каждом этапе жизненного цикла. То есть он рекомендует проводить тестирование на каждом этапе, а не только после этапа реализации, поэтому существует возможность исправления процессов деятельности, а не выделения дополнительных затрат на устранение рисков
- Тестирование - это процесс проверки состояния системы или приложения в соответствии с набором критериев



1. Этап, предшествующий развитию приложения

1.1. Определение жизненного цикла - перед началом разработки приложения должен быть определен соответствующий жизненный цикл, где обеспечение безопасности неизбежно на каждом этапе.

1.2. Анализ политик и стандартов - необходимость обеспечения что существуют политики, стандарты и документация которой необходимо придерживаться в процессе разработки. Документация чрезвычайно важна, поскольку она предоставляет направления и решения для команд разработчиков.

- *«Люди могут делать то что необходимо, только если они знают что делать»*
- *Например, если приложение должно быть разработано на Java, важно иметь безопасный стандарт кодирования Java. Если сформулировано требование о шифровании данных, важно иметь стандарт шифрования*
- *Никакая политика или стандарт не могут охватывать все ситуации, с которыми сталкивается команда разработчиков. Документируя общие и предсказуемые проблемы, будет меньше ситуаций, когда команде придется принимать самостоятельно решения в процессе разработки*

1.3. Разработка показателей (метрик) для обеспечения прослеживаемости - до начала разработки необходимо запланировать измерительную программу. Определяя критерии измерения, видимость дефектов будет обеспечиваться как в процессах, так и в разработанном продукте

2. Определение требований и разработка решения

2.1. Просмотр требований безопасности. Требования безопасности определяют, как приложение работает с точки зрения безопасности. Очень важно, чтобы требования безопасности проверялись. Тестирование в этом случае означает проверку предположений, сделанных в требованиях и тестах, чтобы увидеть, есть ли пробелы в определении требований. Например, если есть требование безопасности, по которому пользователи должны быть зарегистрированы прежде чем они смогут получить доступ к разделу управления сайтом, это будет означать, что пользователь должен предварительно зарегистрироваться или потребовать только аутентификацию (ему был заранее создан аккаунт)? Хорошо, когда требования четко сформулированы и нет нескольких их интерпретаций.

При поиске пробелов в формулировании требований было бы целесообразно обеспечить механизмы обеспечения безопасности, такие как:

- Управление пользователями
- Аутентификация пользователя
- Авторизация пользователя
- Обеспечение конфиденциальности данных
- Обеспечение целостности данных
- Управление сесиями
- Безопасность передачи данных
- Методы разделения системы на подсистемы
- Соблюдение законодательства и соответствие стандартам (включая конфиденциальность и т. д.)

2. Определение требований и разработка решения

2.2. Проектирование архитектуры. Приложения должны иметь хорошо документированный дизайн архитектуры. Эта документация может включать в себя модели, текстовые документы и другие подобные артефакты. Очень важно, чтобы эти артефакты были протестированы, чтобы гарантировать, что дизайн архитектуры предполагает соответствующий уровень безопасности, как определено в требованиях. Идентификация потоков безопасности на этапе проектирования - это не только один из самых экономичных моментов для выявления дефектов, но может быть одним из наиболее эффективных моментов для внесения необходимых изменений. Например, если проект запрашивает авторизацию действий в нескольких местах, может быть целесообразным рассмотреть центральный компонент авторизации. Если приложение выполняет проверку данных в нескольких местах, может оказаться целесообразным разработать централизованный модуль проверки (например, внедрение проверки данных в одном месте, а не в десятке мест намного дешевле). Если обнаружены уязвимости, они должны быть переданы архитектору системы для альтернативных решений

2. Определение требований и разработка решения

2.3. После проектирования архитектуры, необходимо создать и проанализировать модели UML (Unified Modeling Language) – которые описывают логику, как работает приложение. В некоторых случаях они могут быть уже доступны. Эти модели должны использоваться дизайнерами для точного подтверждения работы приложения. Если обнаружены недостатки, они должны быть представлены проектировщику системы для разработки альтернативных решений.

2.4. Создание и анализ моделей рисков для приложения - с архитектурными решениями, UML-моделями, которые объясняют, как работает система, рекомендуется разработать модель рисков, которые угрожают приложению. Необходимо разработать реалистичные сценарии для предотвращения угроз. Следует проанализировать модели проектирования и архитектуры для обеспечения того, чтобы эти угрозы были смягчены, приняты организацией или переданы третьей стороне, например страховой компании. Когда выявленные угрозы не имеют стратегий смягчения, модели проектирования и архитектуры должны быть пересмотрены вместе с системным архитектором с целью изменения моделей

3. Реализация приложения

Теоретически **реализация** - это создание некоторых моделей проектирования, используя определенные технологии. Однако в реальном мире много решений по поводу моделей проектирования принимаются на этапе реализации. Это чаще всего более мелкие решения, или слишком подробные для того что бы описать их на стадии проектирования, или проблемы, при решении которых не был применён какой-то стандарт. Если модели проектирования и архитектуры не правильно созданы или их нет, разработчик должен будет принимать много решений самостоятельно. Если политики и стандартов недостаточно, разработчик столкнется с ситуацией, чтобы принимать еще больше решений.

3.1. Прохождение по коду – команда отвечающая за безопасность должна доработать код, пересмотрев его вместе с разработчиками, а в некоторых случаях и с системными архитекторами. Прокрутка кода - происходит на высоком уровне, с целью объяснения логики и потока реализованного кода. Это позволяет команде отвечающую за проверку кода получить общее представление о коде и позволяет разработчикам объяснить, почему определенные вещи были разработаны так, а не иначе. Целью является не пересмотр кода, а понимание потока, компоновки и структуры кода, составляющего приложение на высоком уровне

3. Реализация приложения

3.2. Редактирование кода - после понимания структуры кода и почему некоторые вещи были закодированы так, как это было сделано, тестер может изучить текущий код с целью обнаружения недостатков безопасности. Статические проверки, проверяют код на основе набора списков проверок, в том числе:

- Требования к доступности, конфиденциальности и целостности
- Рекомендации OWASP или Top 10 для технических воздействий
- Конкретные проблемы, связанные с используемым языком программирования или фреймворком, такими как контрольные списки в PHP или «Microsoft Secure Coding» для ASP.NET;
- Любые требования, специфичные для индустрии программного обеспечения, такие как рекомендации Sarbanes-Oxley 404, COPPA, ISO / IEC 27002, APRA, HIPAA, Visa Merchant и т. д.

Что касается рентабельности от инвестированных ресурсов (в большинстве случаев ресурса времени), тестирование с использованием принципа *white-box testing* дает гораздо более высокую прибыль, чем любой другой метод обзора безопасности, и в меньшей степени зависит от навыков тестировщика. Однако этот метод тестирования не идеален, и код должен быть проверен с точки зрения безопасности, используя и другие более полные методы тестирования

(http://softwaretestingfundamentals.com/white-box-testing/)

4. Внедрение приложения

4.1. Тестирование на проникновение приложений - после тестирования требований, моделирования и анализа кода можно предположить, что все проблемы были зафиксированы. Тестирование приложения после его установки дает окончательную проверку на то, что приложение полностью функционально

4.2. Тестирование управления конфигурацией - тест на проникновение приложения должен включать проверку того, как приложение было развернуто и защищено на уровне сервера. Хотя само приложение может быть защищено, конфигурация может по-прежнему быть уязвимой для эксплуатации

5. Обслуживание

5.1. Разработка отзывов оперативного управления - процесс, подробно описывающий управление как операционной частью приложения, так и инфраструктурой

5.2. Выполнение периодических проверок надежности приложения. Ежемесячные или квартальные проверки должны выполняться как для приложения, так и для инфраструктуры, чтобы гарантировать, что новые угрозы безопасности не были введены и уровень безопасности по-прежнему не поврежден

5.3. Обеспечение проверки изменений. После того, как каждое изменение было одобрено и протестировано, обеспечивая качество приложения, важно, чтобы это изменение было проверено, чтобы гарантировать, что уровень безопасности не изменился

Выводы

- Действия направленные на безопасность веб-приложений в процессе разработки веб-приложений должны распределяться на протяжении всего жизненного цикла
- Рекомендуется учитывать рекомендации OWASP или другие международные и национальные рекомендации и стандарты в процессе разработки веб приложений
- Это обеспечит функциональные, безопасные и надежные веб-приложения