Лабораторная работа 3

«Тестирование программных продуктов. Знакомство со сканерами безопасности»

ЦЕЛЬ РАБОТЫ: изучить классификацию видов тестирования, практически закрепить эти знания путем генерации тестов различных видов, научиться планировать тестовые активности в зависимости от специфики поставляемой на тестирование функциональности.

ОБОРУДОВАНИЕ: ПК, Браузер.

ВРЕМЯ ВЫПОЛНЕНИЯ: 90 минут

КРАТКАЯ ТЕОРИЯ И МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ:

Тестирование (Testing) – процесс анализа программного средства и сопутствующей документации с целью выявления дефектов и повышения качества продукта.

Конечной целью тестирования является предоставление пользователю качественного программного обеспечения (ПО).

Качество (Quality) – степень, с которой компонент, система или процесс соответствует зафиксированным требованиям и/или ожиданиям и нуждам пользователя или заказчика.

Дефект (defect, bug, ошибка) – ключевой термин тестирования, означающий отклонение фактического результата от ожидаемого. Для обнаружения дефекта необходимо выполнить три условия: знать фактический результат, знать ожидаемый результат, зафиксировать факт разницы между фактическим и ожидаемым результатом.

Процесс тестирования как процесс поиска дефектов сводится к следующей последовательности действий:

1. Узнаем ожидаемый результат.
2. Узнаем фактический результат.
3. Сравниваем ожидаемый и фактический результаты.

Источником ожидаемого результата является спецификация – детальное описание того, как должно работать ПО.

В общем случае любой дефект представляет собой отклонение от спецификации. Важно обнаружить эти дефекты до того, как их найдут конечные пользователи.

Тестирование можно классифицировать по очень большому количеству признаков. Далее приведен обобщенный список видов тестирования по различным основаниям.

1. Виды тестирования в зависимости от объекта тестирования: функциональные, пограничные, нефункциональные (рисунок 1).

Рисунок 1 – Классификация видов тестирования в зависимости от объекта

Рассмотрим функциональные виды тестирования.

Функциональное тестирование (Functional Testing) – тестирование, основанное на сравнительном анализе спецификации и функциональности компонента или системы.

Тестирование безопасности (Safety Testing) – тестирование программного продукта с целью определить его способность при использовании оговоренным образом оставаться в рамках приемлемого риска причинения вреда здоровью, бизнесу, программам, собственности или окружающей среде.

Тестирование защищенности (Security Testing) – тестирование с целью оценить защищенность программного продукта от внешних воздействий (от проникновений). На практике зачастую под термином тестирование безопасности понимают в том числе и тестирование защищенности.

Рассмотрим пограничные виды тестирования.

Тестирование совместимости (Compatibility Testing) – проверка работоспособности приложения в различных средах (браузеры и их версии, операционные системы, их типа, версии и разрядность). Виды тестирования совместимости: кроссбраузерное тестирование (различные браузеры или версии браузеров), кроссплатформенное тестирование (различные операционные системы или версии операционных систем).

Рассмотрим нефункциональные виды тестирования, направленные на проверку характеристик или свойств программы (внешний вид, удобство использования, скорость работы и т.п.).

Тестирование требований (Requirements Testing) – проверка требований на соответствие основным атрибутам качества.

Тестирование прототипа (Prototyte Testing) – метод выявления структурных, логических ошибок и ошибок проектирования на ранней стадии развития продукта до начала фактической разработки.

Тестирование пользовательского интерфейса (GUI Testing) – тестирование, выполняемое путем взаимодействия с системой через графический интерфейс пользователя (правописание выводимой информации; расположение и выравнивание элементов GUI; соответствие названий форм / элементов GUI их назначению; унификация

стиля, цвета, шрифта; окна сообщений; изменение размеров окна, поведение курсора и горячие клавиши)

Тестирование удобства использования (Usability Testing) – тестирование с целью определения степени понятности, легкости в изучении и использовании, привлекательности программного продукта для пользователя при условии использования в заданных условиях эксплуатации (на этом уровне обращают внимание на визуальное оформление, навигацию, логичность, наличие обратной связи и др.).

Тестирование доступности (Accessibility Testing) – тестирование, которое определяет степень легкости, с которой пользователи с ограниченными способностями могут использовать систему или ее компоненты.

Тестирование интернационализации (Internationalization Testing) – тестирование адаптации продукта к языковым и культурным особенностям целого ряда регионов, в которых потенциально может использоваться продукт.

Тестирование локализации (Localization Testing) – тестирование адаптации продукта к языковым и культурным особенностям конкретного региона, отличного от того, в котором разрабатывался продукт.

Тестирование производительности (Performance Testing) – процесс тестирования с целью определения производительности программного продукта. В рамках тестирования производительности выделяют нагрузочное тестирование, объемное тестирование, тестирование стабильности и надежности, стрессовое тестирование.

Нагрузочное тестирование (Performance and Load Testing) – вид тестирования производительности, проводимый с целью оценки поведения компонента или системы при возрастающей нагрузке, например количестве параллельных пользователей и/или операций, а также определения какую нагрузку может выдержать компонент или система;

Объемное тестирование (Volume Testing) – позволяет получить оценку производительности при увеличении объемов данных в базе данных приложения;

Тестирование стабильности и надежности (Stability / Reliability Testing) – позволяет проверять работоспособность приложения при длительном (многочасовом) тестировании со средним уровнем нагрузки.

Стрессовое тестирование (Stress Testing) – вид тестирования производительности, оценивающий систему или компонент на граничных значениях рабочих нагрузок или за их пределами, или же в состоянии ограниченных ресурсов, таких как память или доступ к серверу.

Тестирование на отказ и восстановление (Failover and Recovery Testing) – тестирование при помощи эмуляции отказов системы или реально вызываемых отказов в управляемом окружении.

Тестирование установки (Installability Testing) и лицензирования – процесс тестирования установки программного продукта. Включает формальный тест программы установки приложения (проверка пользовательского интерфейса, навигации, удобства использования, соответствия общепринятым стандартам оформления); функциональный тест программы установки; тестирование механизма лицензирования и функций защиты от пиратства; проверку стабильности приложения после установки.

2. Виды тестирования в зависимости от знания кода: белый ящик, серый ящик, черный ящик.

Белый ящик (White Box Testing) – тестирование, основанное на анализе внутренней структуры компонентов или системы (у тестировщика есть доступ к внутренней структуре и коду приложения).

Серый ящик (Grey Box Testing) – комбинация методов белого и черного ящика, состоящая в том, что к части кода архитектуры у тестировщика есть, а к части кода – нет.

Черный ящик (Black Box Testing) – тестирование системы без знания внутренней структуры и компонентов системы (у тестировщика нет доступа к внутренней структуре и коду приложения либо в процессе тестирования он не обращается к ним).

4. Виды тестирования в зависимости от степени автоматизации: ручное, автоматизированное тестирование.

Ручное тестирование – такое тестирование, в котором тест-кейсы выполняются тестировщиком вручную без использования средств автоматизации.

Автоматизированное тестирование (Automated Testing) – набор техник, подходов и инструментальных средств, позволяющий исключить человека из выполнения некоторых задач в процессе тестирования. Тест-кейсы частично или полностью выполняет специальное инструментальное средство.

5. Виды тестирования в зависимости от степени изолированности тестируемых компонентов: модульное, интергационное, системное тестирование.

Модульное тестирование (Unit/Component Testing) – тестируются отдельные части (модули) системы.

Интеграционное тестирование (Integration Testing) – тестируется взаимодействие между отдельными модулями.

Системное тестирование (System Testing) – тестируется работоспособность системы в целом.

6. Виды тестирования в зависимости от подготовленности: интуитивное тестирование, исследовательское тестирование, тестирование по документации.

Интуитивное тестирование выполняется без подготовки к тестам, без определения ожидаемых результатов, проектирования тестовых сценариев.

Исследовательское тестирование – метод проектирования тестовых сценариев во время выполнения этих сценариев.

Тестирование по документации – тестирование по подготовленным тестовым сценариям, руководству по осуществлению тестов.

7. Виды тестирования в зависимости от места и времени проведения тестирования: приемочное тестирование, альфа-тестирование, бета-тестирование.

Приемочное тестирование (User Acceptance Testing, UAT) – формальное тестирование по отношению к потребностям, требованиям и бизнес процессам пользователя, проводимое с целью определения соответствия системы критериям приёмки и дать возможность пользователям, заказчикам или иным авторизованным лицам определить, принимать систему.

Альфа-тестирование (Alpha Testing) – моделируемое или действительное функциональное тестирование, выполняется в организации, разрабатывающей продукт, но не проектной командой (это может быть независимая команда тестировщиков, потенциальные пользователи, заказчики). Альфа тестирование часто применяется к коробочному программному обеспечению в качестве внутреннего приемочного тестирования.

Бета-тестирование (Beta Testing) – эксплуатационное тестирование потенциальными или существующими клиентами/заказчиками на внешней стороне (в среде, где продукт будет использоваться) никак связанными с разработчиками, с целью определения действительно ли компонент или система удовлетворяет требованиям клиента/заказчика и вписывается в бизнес-процессы.

Бета-тестирование часто проводится как форма внешнего приемочного тестирования готового программного обеспечения для того, чтобы получить отзывы рынка.

8. Виды тестирования в зависимости от глубины тестового покрытия: Smoke, MAT, AT.

Тестовое покрытие – одна из метрик оценки качества тестирования, представляющая из себя плотность покрытия тестами требований либо исполняемого кода.

Smoke Test – поверхностное тестирование для определения пригодности сборки для дальнейшего тестирования, должно покрывать базовые функции программного обеспечения; уровень качества:

Acceptable / Unacceptable. Minimal Acceptance Test (MAT, Positive Test) – тестирование системы или ее части только на корректных данных/сценариях; уровень качества: High / Medium / Low.

Acceptance Test (AT) – полное тестирование системы или ее части как на корректных (Positive Test), так и на некорректных данных/сценариях (Negative Test); уровень качества: High / Medium / Low. Тест на этом уровне покрывает все возможные сценарии тестирования: проверку работоспособности модулей при вводе корректных значений; проверку при вводе некорректных значений; использование форматов данных отличных от тех, которые указаны в требованиях; проверку исключительных ситуаций, сообщений об ошибках; тестирование на различных комбинациях входных параметров; проверку всех классов эквивалентности; тестирование граничных значений интервалов; сценарии не предусмотренные спецификацией и т.д.

9. Виды тестирования в зависимости от тестовых активностей: NFT, RT, DV.

Данная классификация тестирования иначе назвыется видами тестирования в зависимости от ширины тестового покрытия.

Тестирование новых функциональностей (New Feature Test, NFT) – определение качества поставленной на тестирование новой функциональности, которая ранее не тестировалась. Данный тип тестирования включает в себя: проведение полного теста (AT) непосредственно новой функциональности; тестирование новой функциональности на соответствие документации; проверку всевозможных взаимодействий ранее реализованной функциональности с новыми модулями и функциями.

Регрессионное тестирование (Regression Testing, RT) проводится с целью оценки качества ранее реализованной функциональности. Включает в себя проверку стабильности ранее реализованной функциональности после внесения изменений, например добавления новой функциональности, исправление дефектов, оптимизация кода, разворачивание приложения на новом окружении.

Регрессионное тестирование как правило выполняется на уровне MAT.

Валидация дефектов (Defect Validation, DV) – проверка результатов исправления дефектов; может включать элементы регрессионного тестирования; уровень проверки не определяется.

Процесс тестирования программного продукта включает следующие этапы:

1. Изучение и анализ предмета тестирования.
2. Планирование тестирования.
3. Исполнение тестирования.

Изучение и анализ предмета тестирования начинается еще до утверждения спецификации и продолжается на стадии разработки (кодирования) программного обеспечения. Конечной целью этапа изучения и анализа предмета тестирования является получение ответов на два вопроса: какие функциональности предстоит протестировать, как эти функциональности работают.

Планирование тестирования происходит на стадии разработки (кодирования) программного обеспечения. На стадии планирования тестирования перед тестировщиком стоит задача поиска компромисса между объемом тестирования, который возможен в теории, и объемом тестирования, который возможен на практике. На данной стадии необходимо ответить на вопрос: как будем тестировать? Результатом планирования тестирования является тестовая документация.

Выполнение тестирования происходит на стадии тестирования и представляет собой практический поиск дефектов с использованием тестовой документации, составленной ранее.

Для всех программных продуктов выполняют следующие типы тестов и их композиции.

Для первой поставки программного обеспечения рекомендуется проводить Smoke + NFTAT готовой функциональности: поверхностное тестирование (Smoke Test)

выполняется для определения пригодности сборки для дальнейшего тестирования; полное тестирование системы или ее части как на корректных, так и на некорректных данных/сценариях (Acceptance Test, AT) позволяет обнаружить дефекты и внести запись о них в багтрэкинговую систему.

Для последующих поставок программного обеспечения композиции тестов могут быть следующими.

Если не была добавлена новая функциональность, то: DV + RTMAT. Т.е., выполняется проверка исправления дефектов программистом (Defect Validation, DV), а также проверка работоспособности остальной функциональности после исправления дефектов на позитивных сценариях (Minimal Acceptance Test, MAT).

Если была добавлена новая функциональность, то:

Smoke + DV + NFTAT + RTMAT. В частности, выполняется поверхностное тестирование (Smoke Test), проверка исправления дефектов программистом (Defect Validation, DV), тестирование новых функциональностей (New Feature Testing, NFT), проверка старых функциональностей, т.е. регрессионное тестирование (Regression Test).

Если была добавлена новая функциональность, то возможен также вариант: DV + NFTAT + RTMAT, т.е. без выполнения Smoke Test. Таким образом, для второй и последующих поставок обобщенная схема композиции тестов выглядит следующим образом:

(Smoke) + DV + (NFTAT) + RTMAT. В зависимости от типа и специфики приложения (web, desktop, mobile) выполняют специализированные тесты (например, кроссбраузерное или кроссплатформенное тестирование, тестирование локализации и интернационализации и др.).

Сканеры безопасности

Если вы размещаете свои веб-приложения на администрируемом сервере или шаред хостинге, тогда Вам не о чем беспокоиться. Однако для виртуального или выделенного сервера вы должны предусматривать все возможности для безопасности своего сервера. Иметь ненужные порты открытыми — плохая идея, из которой злоумышленник может извлечь выгоду множеством способов.

Ниже представлены бесплатные онлайн сервисы, которые помогут выяснить открыты ли порты, чтобы вы могли проверить и заблокировать их, если они не используются.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ И ФОРМА ОТЧЕТНОСТИ:

Задание

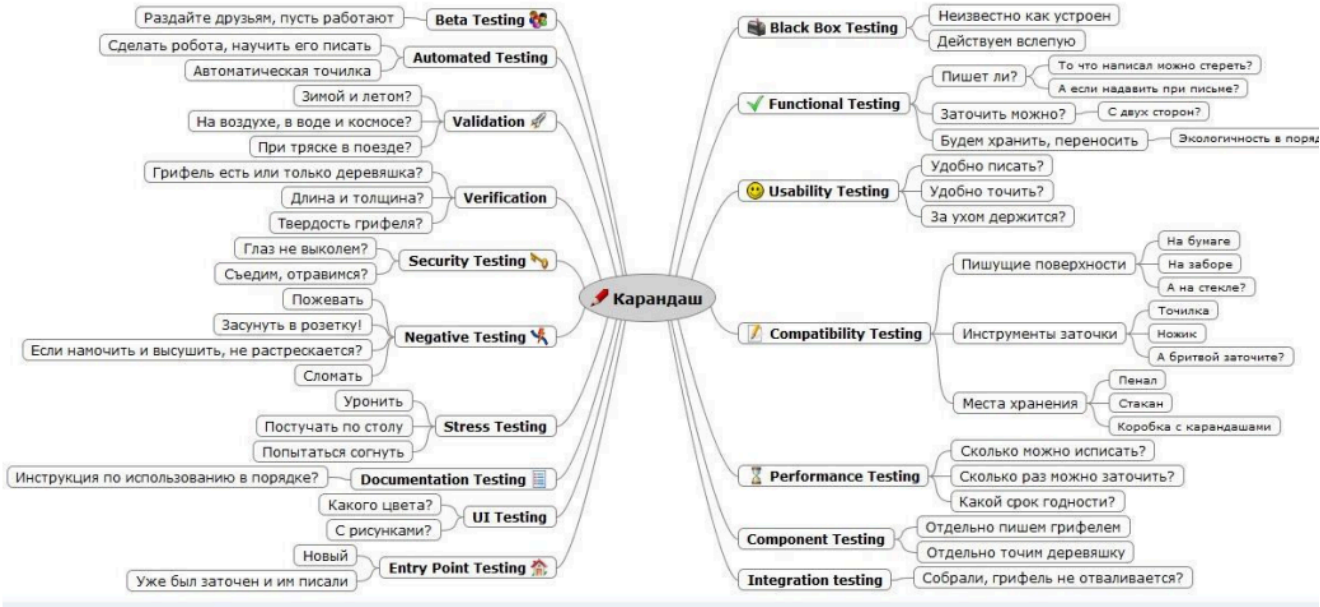
1. Выбрать объект реального мира (например, карандаш, стол, чашка, клавиатура, сумка и др.) с целью последующей разработки тестовых проверок для него.
2. Разработать различные проверки в соответствии с классификацией видов тестирования для выбранного объекта реального мира. Результаты внести в таблицу.

(пример приведен на рисунке 1,
краткое описание видов тестирования приведено в теоретической части).

Рисунок 1

Пример выполнения лабораторной работы

Необходимо составить тестовый план для объекта «Карандаш».
Пример тестового плана для объекта карандаш представлен на рисунке 1.1.



Объект тестирования: *указать*		
Вид тестирования	**Краткое определение вида тестирования**	**Тестовые проверки**
Functional Testing		
Safety Testing		
Security Testing		
Compatibility Testing		
GUI Testing		
Usability Testing		
Accessibility Testing		
Internationalization Testing		
Performance Testing		

Stress Testing		
Negative Testing		
Black Box Testing		
Automated Testing		
Unit/Component Testing		
Integration Testing		

Задание 2

С помощью утилиты ping (в командной строке) узнать IP-адрес любого интересующего вас сайта.

Задание 3

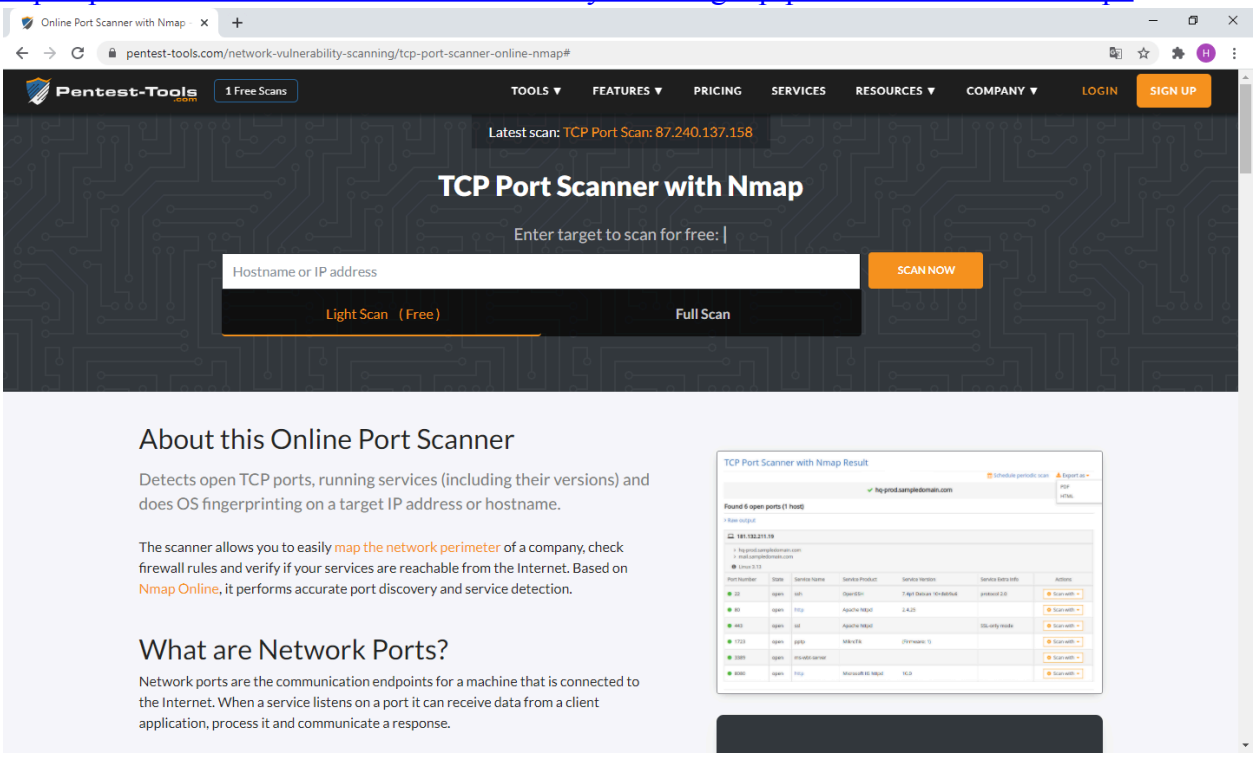
TCP сканер портов с Nmap

Pentest Tools проверяет открытые порты, используя NMAP на целевой хост. Вы можете выбрать определение операционной системы и версию сервиса.

Перейдя по следующей ссылке вам необходимо в строку «Hostname or IP address» ввести IP-адрес сайта, полученный в прошлом задании. Учтите, что проверку можно проводить не больше 2х раз в сутки для одного незарегистрированного пользователя.

После того, как выбранный сайт будет просканирован, скачайте отчет (Download report) и ознакомьтесь с ним. В отдельный документ вынесете найденные уязвимости и расшифруйте.

https://pentest-tools.com/network-vulnerability-scanning/tcp-port-scanner-online-nmap#



Задание 4

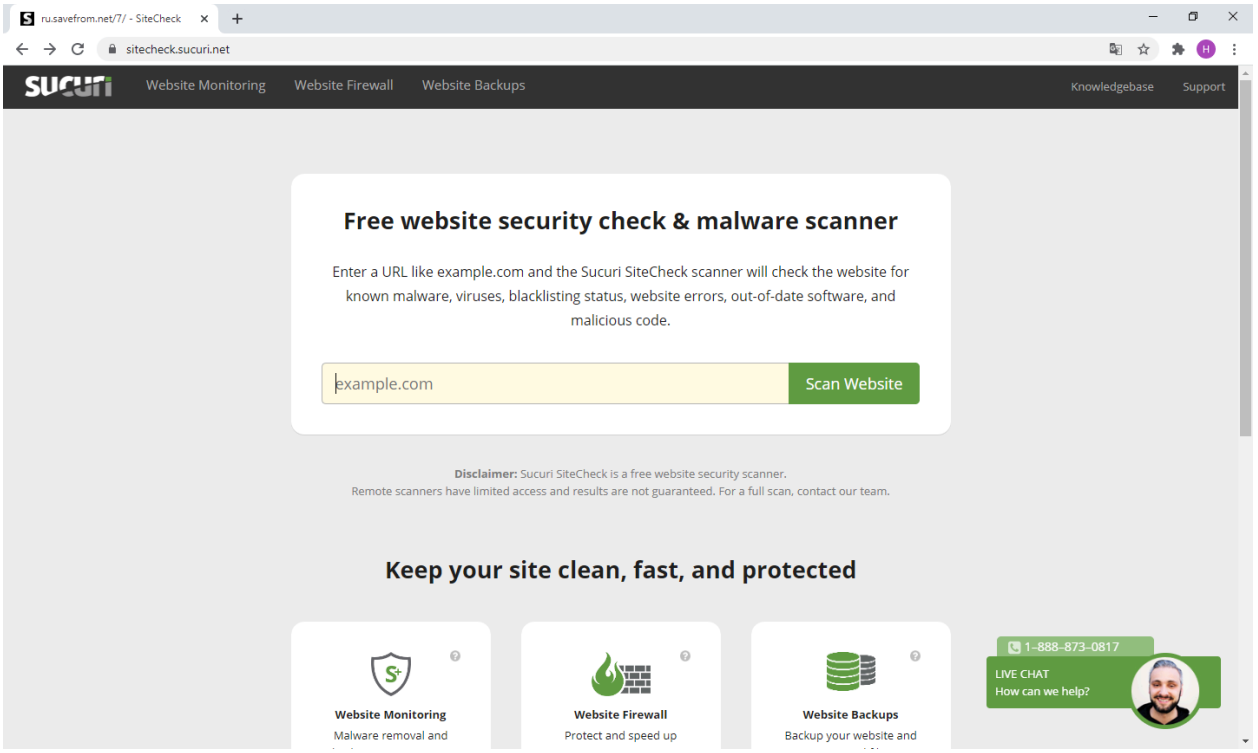
Онлайн URL сканер для проверки вирусов на сайте

VirusTotal не является в полной мере веб-сканером, но его рекомендуется использовать для диагностики, так как он является агрегатором нескольких десятков антивирусов и антивирусных сервисов.

https://www.virustotal.com/gui/home/url

Выберите любой сайт (желательно, потенциально вредоносный) и проверьте его с помощью данного сайта. В качестве отчета о проделанной работе приложите скриншоты результатов сканирования.

Задание 5



https://sitecheck.sucuri.net/

Таким же образом, как и в предыдущем задании, проверить любой выбранный Вами сайт. Результаты проверки вывести в отдельный документ и расшифровать найденные риски.

Задание 6

Онлайн-сканер от UpGuard показывает анализ безопасности в формате геймификации. После сканирования сайт получает определенное количество баллов, в зависимости от следующих факторов:

- установлен ли у вас SSL-сертификат;
- есть ли защита доменного имени;
- замечено ли на сайте вредоносное ПО;
- открытая информация о сервере;
- включена ли SPF (Sender Policy Framework, инфраструктура политики отправителя);
- и многое другое.

https://app.upguard.com/webscan

Провести сканирование сайта, выделить и оценить обнаруженные уязвимости.

Задание 7

Сравнить все отчеты, полученные в предыдущих заданиях. Выделить наиболее часто повторяющиеся уязвимости веб-ресурсов. Написать вывод по полученным данным.

Контрольные вопросы:

1. Что такое тестирование методом черного ящика?
2. Что такое тестирование белым ящиком?
3. Как выполняется тестирование логики программы?
4. Как выполняется тестирование данных в программе?
5. Что такое сканер безопасности?
6. Приведите примеры сканеров безопасности.

7. Какие виды сканеров уязвимости можно выделить?
8. При помощи каких двух два основных механизмов сканер проверяет наличие уязвимости?
9. Опишите этапы сканирования.