Система контроля защищенности интернет-ресурсов (сайтов) – Веб Безопасность

Описание программного обеспечения

Аннотация

Настоящий документ содержит необходимые сведения для изучения программного обеспечение системы контроля защищенности интернет-ресурсов (сайтов) – Веб Безопасность (далее – ПО).

В руководстве содержатся сведения о назначении, составе и технических характеристиках ПО, сведения об устройстве и работе, порядок использования ПО по назначению.

Для изучения и правильной технической эксплуатации ПО требуется персонал, обладающий знаниями и навыками работы с ЭВМ, техническими средствами вычислительных сетей и прошедший курс обучения работе с ПО.

1. Описание и работа ПО

1.1. Назначение ПО

Полное наименование: ПО «Система контроля защищенности интернет-ресурсов (сайтов) – Веб Безопасность».

Сокращенное наименование: ПО «Webbez Scanner».

Перед эксплуатацией ПО необходимо внимательно ознакомиться с комплектом эксплуатационной документации.

В случае обнаружения дефектов технических и программных средств следует обращаться к поставщику ПО.

Настоящий документ должен находиться у должностного лица, ответственного за эксплуатацию ПО.

ПО системы контроля защищенности интернет-ресурсов (сайтов) – Веб Безопасность представляет собой набор решений, направленных на обеспечение функций контроля защищенности информационной системы (далее - ИС), построенной на базе веб-технологий. Использование данной системы позволяет реализовать функции обнаружения, предотвращения и реагирования на попытки несанкционированной активности удаленных и внутренних пользователей компьютерных сетей.

В ПО реализованы следующие области проверки защищенности ИС, построенных на базе веб-технологий:

- предварительный сбор информации;
- использование поисковых систем для поиска чувствительных данных в информационном пространстве ИС;
- проведение конфигурационного анализа;
- проведение тестов на наличие типовых уязвимостей ИС на базе веб-технологий;
- проверка наличия недостатков в используемых механизмах аутентификации;
- проведение комплексных проверок безопасности используемых веб-технологий;
- поиск стороннего кода;
- выявление фактов непреднамеренного раскрытия чувствительной информации.

1.2. Состав программного обеспечения

ПО содержит следующие основные функциональные модули:

– веб-интерфейс предназначен для постановки задач и просмотра сгенерированных отчетов пользователем;

– робот – наименьшая структурная единица. Робот представляет собой программный модуль, направленный на выполнение проверки конкретного аспекта безопасности ИС;

– механизм контроля расписания и запуска локальных менеджеров;

– модуль постоянного контроля, который предназначен для периодических проверок заданных ИС;

– демон Daemon (локальный менеджер модуля «Постоянный контроль»), который обеспечивает запуск новых заданий на сканирование, контроль корректности выполнения ранее запущенных заданий, корректировку глобального расписания, запуск механизма формирования конечного отчета;

– механизм формирования конечного отчета применяется в модуле постоянного контроля для предоставления пользователю результатов в форме pdf-файла;

– база данных MSQL предназначена для хранения данных о доступных роботах,

выполняемых проверках, зарегистрированных пользователях и исследуемых ИС;

– робот ar_checker представляет собой программное решение для интеграции с фреймворком Arachni;

– модуль «Экспресс», который предназначен для разовых проверок заданных ИС;

– API-интерфейс модуля «Экспресс» представляет собой полноценный REST-API, который позволяет осуществлять постановку задач и получение результата в составе интегрированных программных комплексов;

– демон apiDaemon (локальный менеджер модуля «Экспресс»), который обеспечивает запуск новых заданий на сканирование, контроль корректности выполнения ранее запущенных заданий, запуск механизма формирования конечного отчета;

– база данных SQLT предназначена для хранения данных о выполняемых проверках, статусе поставленных задач, доступных пакетах модуля экспресс контроля.

1.3. Описание архитектуры ПО

ПО с позиции внутреннего устройства представляет собой набор независимых модулей, подсистем и интерфейсов, которые предназначены для решения строго определенных задач. Взаимодействие между элементами комплекса происходит через типовые узловые механизмы: базы данных, текстовые файлы и api-интерфейсы. Взаимодействие происходит без использования системных механизмов операционной системы, что снижает трудозатраты при анализе нештатных ситуаций. Общий вид архитектуры ПО представлен на рисунке 1.

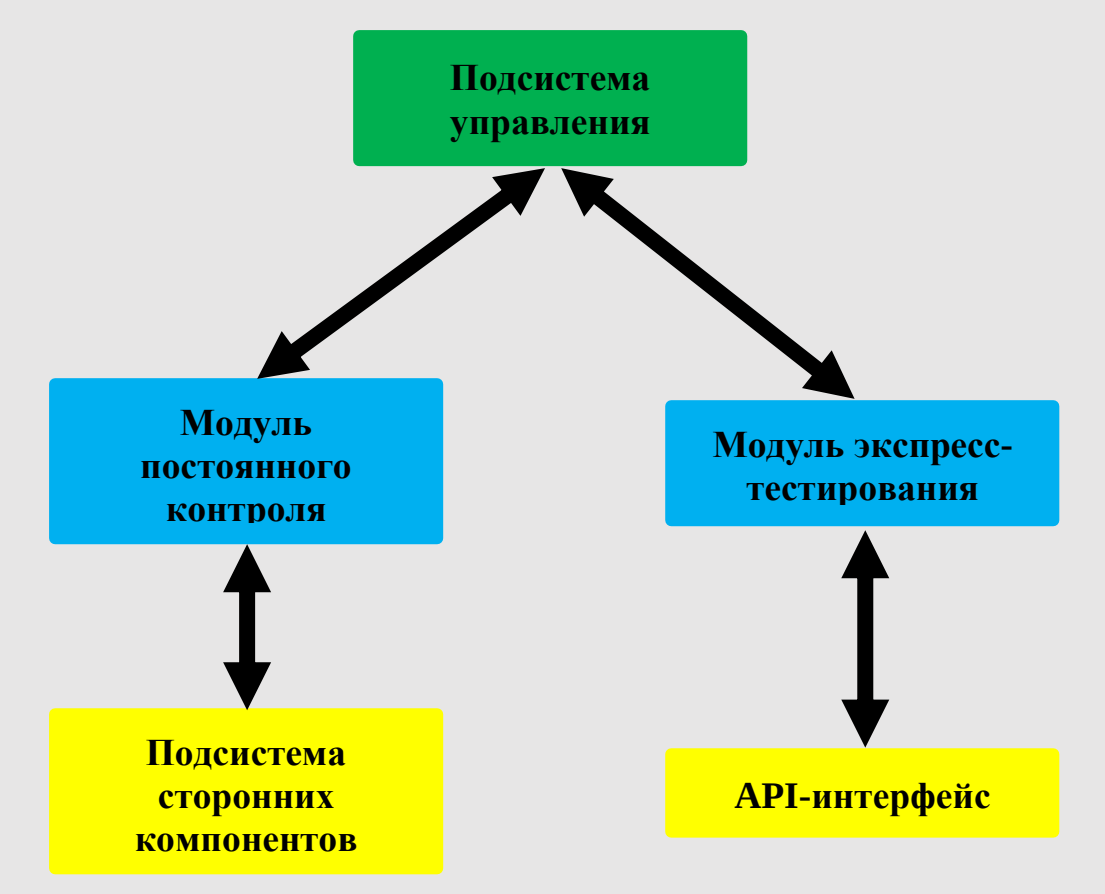

Рисунок 1. Верхнеуровневая архитектура ПО

Представленные элементы выполняют следующие функции и задачи:

«Подсистема управления» представляет собой решение на базе веб-приложения и

предназначена для постановки задач и просмотра сгенерированных отчетов пользователем;

«Модуль постоянного контроля» предназначен для выполнения периодических сканирований заданных веб-ресурсов с последующей генерацией отчетных материалов в форме pdf-отчета и html-версии;

«Модуль экспресс-тестирования» предназначен для выполнения разовых сканирований заданных веб-ресурсов с последующей генерацией html-версии отчета;

«Подсистема сторонних компонентов» предназначена для расширения перечня выполняемых типов сканирования с помощью свободно распространяемых программных библиотек;

«API-интерфейс» представляет собой полноценный REST-API, который позволяет осуществлять постановку задач и получение результата в составе интегрированных программных комплексов.

1.3.1. Подсистема управления

«Подсистема управления» представляет собой решение на базе веб-приложения. Непосредственно веб-приложение разработано на базе PHP-фреймворка CodeIngnitor. Данная подсистема решает задачи по управлению ПО, а также предоставляет возможность для постановки задач на сканирование веб-ресурсов и просмотра сгенерированных отчетов пользователем.

Общий вид подсистемы управления представлен на рисунке 2:

Рисунок 2. Подсистема управления ПО

«Раздел пользователя» и «Раздел администрирования» логически разделены друг от друга с помощью внутренних механизмов PHP-фреймворка CodeIgnitor, что позволяет гибко настроить политику доступа к ним как на прикладном, так и на сетевом уровнях.

1.3.2. Модуль постоянного контроля

«Модуль постоянного контроля» предназначен для выполнения периодических сканирований заданных веб-ресурсов с последующей генерацией отчетных материалов в форме pdf-отчета и html-версии. При проектировании и разработке данного компонента ПО также был сделан акцент в сторону выделения отдельных функциональных элементов в самостоятельные и независимые сущности. Это позволило достаточно оперативно вносить изменения в логику

работы всего модуля, а также добавлять новые возможности.

Общая схема «модуля постоянного контроля» представлена на рисунке 3:

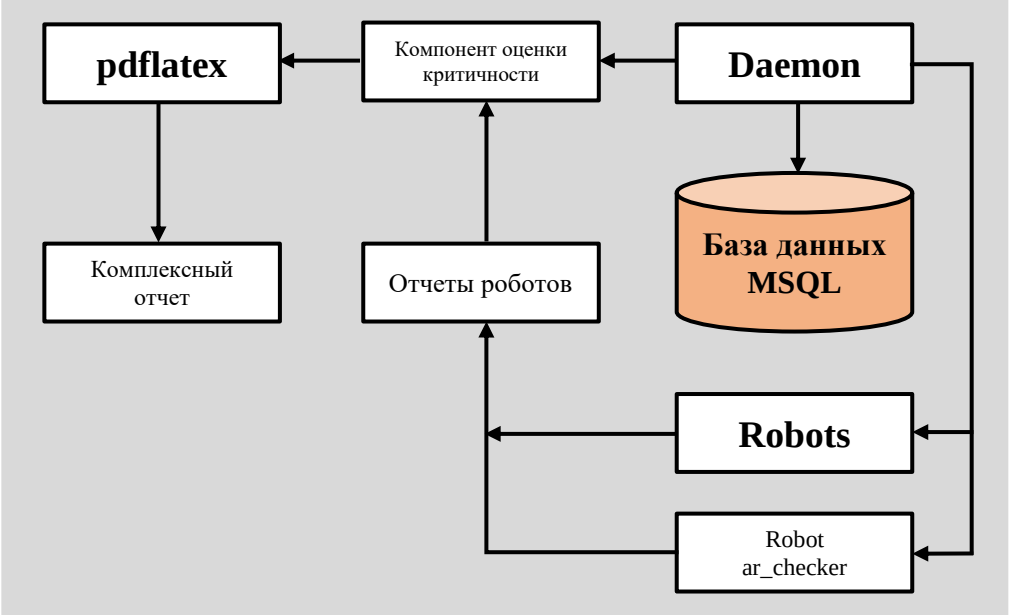

Рисунок 3. Модуль постоянного контроля ПО

Представленные элементы выполняют следующие функции и задачи:

1) Демон Daemon является менеджером всего модуля и решает следующие задачи: обеспечивает запуск новых заданий на сканирование, контроль корректности выполнения ранее запущенных заданий, корректировку глобального расписания, запуск механизма формирования конечного отчета;

2) База данных MSQL предназначена для хранения данных о доступных роботах, выполняемых проверках, зарегистрированных пользователях и исследуемых ИС;

3) Роботы – наименьшая структурная единица. Роботы представляют собой программные модули, направленные на выполнение проверки конкретного аспекта безопасности ИС. Архитектура модуля позволяет включать в общий список активных проверок роботов реализованных на наиболее популярных языках программирования;

4) Робот ar_checker – универсальный робот для взаимодействия с API программного фреймворка Arachni;

5) Отчеты роботов – xml-файлы, в которых содержится информации о времени выполнения проверки, полученных результатов, а также выставленном уровне критичности для обнаруженных уязвимостях;

6) Компонент оценки критичности определяет общий уровень угрозы для проверяемого веб-ресурса на основе уровня критичности всех выполненных проверок;

7) pdflatex – стороннее и свободно-распространяемое программное решение для генерации pdf-отчетов;

8) Комплексный отчет - механизм формирования конечного отчета для предоставления пользователю результатов в форме pdf-файла и html-версии.

1.3.3. Модуль экспресс-тестирования

«Модуль экспресс-тестирования» предназначен для выполнения разовых сканирований заданных веб-ресурсов с последующей генерацией html-версии отчета. Перечень проверок и режим работы роботов аналогичен модулю постоянного контроля.

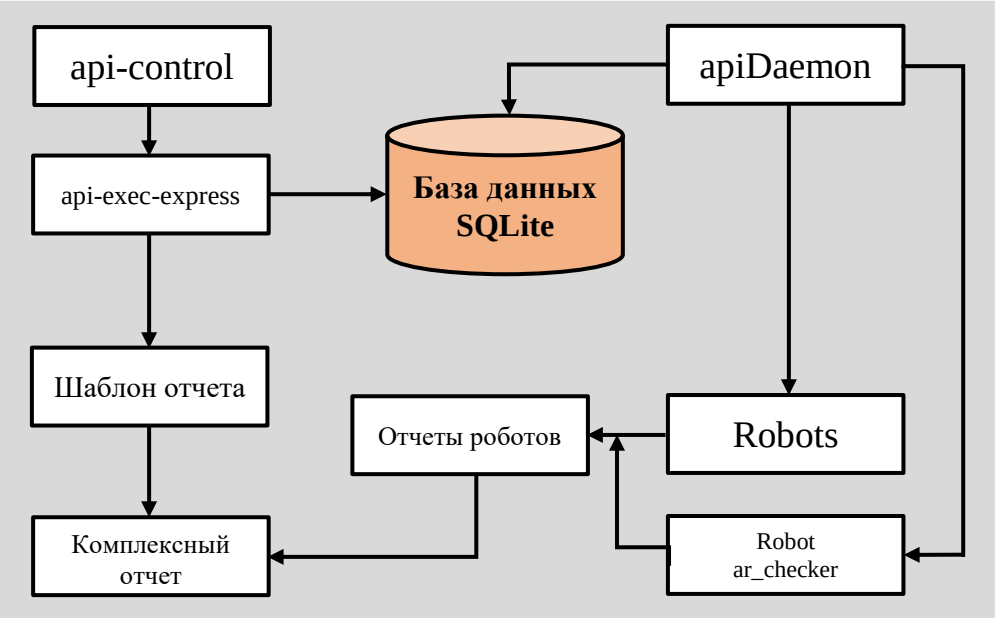

Рисунок 4. Модуль экспресс-тестирования ПО

Представленные на рисунке 4 элементы выполняют следующие функции и задачи:

1) Демон apiDaemon обеспечивает запуск новых заданий на сканирование, контроль корректности выполнения ранее запущенных заданий, запуск механизма формирования комплексного отчета;

2) Роботы – наименьшая структурная единица. Роботы представляют собой программные модули, направленные на выполнение проверки конкретного аспекта безопасности ИС. Архитектура модуля позволяет включать в общий список активных проверок роботов реализованных на наиболее популярных языках программирования;

3) Робот ar_checker представляет собой программное решение для интеграции с фреймворком Arachni;

4) api-control – управляющий элемент в составе API-интерфейса ПО. Выполняет задачи по контролю доступа к API на прикладном уровне, предоставляет список доступных исполняемых модулей, транслирует в исполнительный элемент задачи по добавлению новых сканирований, а также на получение сформированных отчетов;

5) api-exec-express – исполнительный элемент в составе API-интерфейса ПО. Выполняет задачи по последовательному запуску проверок, соответствующих выбранному режиму сканирования, обновляет статус по выполненным работам, формирует шаблон для запущенного исследования;

6) Шаблон отчета – генерируемый исполняемый файл для каждого нового сканирования заданного веб-ресурса. После завершения работы всех роботов шаблон позволяет объединить полученные результаты, разделенные на соответствующие категории;

7) Отчеты роботов – xml-файлы, в которых содержится информации о времени выполнения проверки, полученных результатов, а также выставленном уровне критичности для обнаруженных уязвимостях;

8) Комплексный отчет - механизм формирования конечного отчета для предоставления пользователю результатов в форме pdf-файла и html-версии.

1.3.4. API-интерфейс

API-интерфейс ПО представляет собой полноценный REST-API, который позволяет осуществлять постановку задач и получение результата в составе интегрированных программных комплексов.

Архитектура этого компонента практически полностью повторяет архитектуру «модуля экспресс-тестирования» (см. раздел «Модуль экспресс-тестирования» данного руководства). Точнее будет сказать, что модуль экспресс-тестирования представляет собой реализацию работы с API-интерфейсом.

API-интерфейс поддерживает следующие методы:

– методы контроллера /serverexec
– /status – получить информацию о состоянии сервера
– /action – получить информацию о списке возможных мероприятий сервера
– методы контроллера /task
– /addtask – добавить новую задачу
– /deletetask – удалить ранее поставленную задачу
– /statustask – получить информацию о состоянии ранее поставленной задачи
– /reporttask – получить отчет о ранее поставленной задачи

1.3.5. Подсистема сторонних компонентов

«Подсистема сторонних компонентов» предназначена для расширения перечня выполняемых типов сканирования с помощью свободно распространяемых программных библиотек. На рисунке 5 представлена логическая схема данного элемента в ПО.

Рисунок 5. Подсистема сторонних компонентов ПО

В текущей версии «Подсистема сторонних компонентов» представлена следующими элементами:

Ядро Arachni – основной исполнительный механизм свободно-распространяемого фреймворка для тестирования веб-приложений. При его внедрении в ПО были проведены дополнительные работы по изучению режимов его работы. На основании полученных результатов были оптимизированы варианты его использования для снижения нагрузки на проверяемый веб-ресурс, а также сокращению времени его функционирования;

API-интерфейс фреймворка Arachni – серверный компонент фреймворка, который запускается вместе с операционной системой, на которой работает ПО. Он предоставляет возможность отправки заданий на сканирование и получение готовых результатов.

1.4. Функциональные возможности

1.4.1. Сбор информации

- составление карты сайта
- получение статусов всех обнаруженных директорий сайта
- проверка активированных http-методов

1.4.2. Работа с поисковыми системами

- поиск нежелательных данных сайта в поисковой системе Google
- мониторинг упоминания сайта на хакерских платформах

1.4.3. Конфигурационный анализ

- систем администрирования
- распространенных backdoor-файлов
- резервных копий исполняемых модулей
- административных и внутренних дочерних доменов
- доступных систем управления версиями CVS/SVN, GIT, Mercurial
- директорий без индексного файла (листинг директории)
- небезопасного использования cookies
- небезопасного использования технологии CORS
- небезопасного использования кросс-доменной политики
- небезопасного использования политики клиентского доступа
- наличие служебных объектов (по словарю)
- наличие и настройку WebDAV
- защищенность персональных данных
- анализ файла robots.txt
- цифровые сертификаты
- использование политики «Strict-Transport-Security»
- использование защитного механизма «X-Frame-Options»

1.4.4. Проверка уязвимостей

- XPath-инъекции
- LDAP-инъекции
- SQL-инъекции
- SQL-инъекции на основе появления ошибок
- NoSQL-инъекции на основе появления ошибок
- «слепых» SQL-инъекции на основе дифференциального анализа
- «слепых» SQL-инъекции на снове анализа временных издержек
- «слепых» NoSQL- инъекции на основе дифференциального анализа
- инъекции исполняемого кода
- «слепые» инъекции исполняемого кода на основе появления ошибок
- инъекции команд хостовой ОС
- «слепые» инъекции команд хостовой ОС на основе анализа временных издержек
- раскрытие исходного кода
- возможностей для несанкционированного подключения файлов
- возможностей для CRF-атаки
- возможностей для обхода директорий
- возможностей для несанкционированного подключения удаленных файлов
- возможностей для «расщепления» ответов сервера

- возможностей для неконтролируемого перенаправления пользователя
- возможностей для неконтролируемого DOM-перенаправления пользователя
- возможностей для внедрения XSS-векторов
- возможностей для внедрения XSS-векторов в URL
- возможностей для внедрения XSS-векторов в атрибуты html-элементов
- возможностей для внедрения XSS-векторов в html-элементы
- возможностей для внедрения XSS-векторов в содержимое пользоваетльских скриптов
- возможностей для внедрения XSS-векторов в DOM-модель страницы
- возможностей для атаки «XML External Entity»
- проверка типовых паролей для basic-авторизации (по словарю)
- анализ защищенности модулей аутентификации
- определение плагинов и наличия уязвимостей в популярных CMS
- поиск сайта в различных «черных» списках
- проверка приложения устойчивости к DoS-атакам
- поиск следов вирусов
- поиск документов без прямых ссылок на сайте

1.4.5. Возможность многопользовательской работы, и разграничение прав доступа к проектам.

1.5. Базовые методы проверок

В ПО присутствует набор базовых сканеров, направленных на комплексный сбор предварительной информации о ресурсе, их результаты используются в дальнейших проверках.

1) Составление карты исследуемого сайта

Алгоритм направлен на получение содержимого страниц исследуемого сайта и обход ссылок веб-сайта, начиная с «главной страницы». По сути, этим алгоритмом пользуются все поисковые машины (Google, Yandex и т.д.). Робот расходует меньше ресурсов исследуемого сайта, так как переход осуществляется по ограниченному количеству ссылок. Содержимое страниц накапливается в локальном хранилище, что минимизирует повторное обращение последующими роботами.

2) Получение статусов всех обнаруженных директорий сайта

После построение основной карты сайта следующий робот производит ранжирование всех обнаруженных ссылок, отбрасывает названия конечных html-страниц и производит обращения к корню директорий, в которых расположены html-страницы. Так производится базовая проверка на наличие информационных утечек.

3) Проверка наличия служебных объектов (по словарю)

В составе ПО присутствует специальный словарь, который содержит названия административных и служебных модулей, которые наиболее часто используют разработчики для создания административных панелей управления. Текущий робот проверяет их наличие и права доступа к ним. При обнаружении, соответствующие пути и названия передаются в конечный отчет.

4) Поиск директорий без индексного файла (листинг директории)

На данном этапе проверяется наличие недостатков в администрировании сайта, при которых стороннему пользователю доступен список всех файлов в проверяемой директории, которая может содержать файлы ограниченного доступа.

5) Проверка типовых паролей для basic-авторизации (перебор по словарю)

Если на предыдущих этапах сканирования были обнаружены директории, в которых доступ осуществляется с использованием basic-авторизации, то текущий робот осуществляет проверку на использование типовых паролей и имен пользователей (брутфорс). Количество проверок не превышает 200. При обнаружении параметров авторизации происходит оповещение владельца ресурса, а данные заносятся в отчет.

6) Поиск нежелательных данных в Google, проиндексированных на сайте

В результате предварительных исследований был составлен список специальных обращений к поисковой системе Google, которые показывают наличие проиндексированных документов, имеющих отношение к возможным утечкам информации. Общее название технологии широко известно как «Google-хакинг». ПО использует собственную базу запросов.

7) Поиск следов установленных вирусов

На первых этапах исследования роботы получают содержимое практически всех страниц исследуемого веб-ресурса. Текущий робот производит их анализ на наличие специальных программных вкладок, которые могут нанести вред посетителям интернет-ресурса. Их появление в большинстве случаев связано с получением контроля над сайтов злоумышленниками (взлом сайта, взлом провайдера, инсайд). В составе ПО имеется богатый набор сигнатур, которые используют злоумышленники для встраивания собственного кода.

8) Проверка наличия SQL-инъекций

Как правило, уязвимости в модулях интернет-ресурса возникают из-за неправильной обработки данных пользователя. Текущий робот, используя карту сайта, полученную ранее, осуществляет передачу как "ожидаемых" значений параметров, так и спецсимволов. После этого происходит сравнение ответов от веб-сервера и выделяются значения, которые могут оказывать негативное влияние на модуль интернет-ресурса.

Вероятность нанести такими действиями непоправимый вред веб-серверу равна нулю. Все возникающие ошибки относятся только к действующему сеансу и не затрагивают остальных пользователей.

По результатам данного этапа составляется список модулей интернет-ресурса, которые могут служить источниками утечек информации.

9) Поиск резервных копий исполняемых модулей

Типичная ошибка администрирования - это одновременное хранение на сервере различных версий модулей сайта, используя для этого угадываемые названия и сокращения. Зная перечень всех модулей исследуемого интернет-ресурса, на текущем этапе строится список возможных названий файлов и осуществляется проверка их существования.

Основная опасность хранения различных версий - это дополнительные ошибки и отладочные сообщения, которые могут служить источником утечки информации.

10) Определение плагинов и наличия уязвимостей в популярных CMS

Около 35% всех интернет-ресурсов построено на платформе Joomla, Wordpress или Drupal (согласно исследованиям 2016 года). Уязвимости данных платформ автоматически делают уязвимыми и интернет-ресурсы, которые их используют.

На данном этапе устанавливается тип и версия используемого движка, а также наличие дополнительных функциональных расширений, которые также могут содержать уязвимости. После этого по базе проверяется наличие обнаруженных уязвимостей и формируется отчет. Так как использование этих движком чрезвычайно распространено, то зная недостатки всего лишь одного из расширений, становится возможным в короткое время получить контроль над множеством интернет-ресурсов.

11) Поиск административных и внутренних дочерних доменов

Как правило, крупные организации не ограничиваются использованием одного доменного имени. Для решения специальных задач, таких как веб-почта, площадка для ведения разработки, архив документов для внутреннего пользования и т.д. Основные средства по защите направлены на главный веб-сайт организации, что делает дочерние поддомены более уязвимыми. Часто в них содержится наиболее ценная информация.

Текущий робот проверяет наличие наиболее распространенных дочерних доменов для исследуемого сайта и предоставляет полный список для возможного дальнейшего исследования.

12) Мониторинг упоминания сайта на хакерских платформах

Достаточно часто на хакерских площадках происходит обсуждение найденных информационных утечек, способы обхода защиты и просьбы о помощи для взлома.

Текущий робот производит мониторинг этих площадок и в случае появления упоминаний о целевом ресурсе исследования записывает событие в отчет.

13) Проверка цифровых сертификатов

Методика для проверки цифровых сертификатов сайтов, которые используют защищенное соединение (https).

Получаемые сведения:

- кто и для кого выдал сертификат;
- дата получения и окончания;
- алгоритм и длина ключа.

Если сертификат заканчивается, то выдается предупреждение.

14) Анализ файла robots.txt

Робот осуществляет поиск и анализ файла robots.txt. При его обнаружении производится детальный разбор встроенных директив и выводится список разделов интернет-ресурса, запрещенных для индексирования поисковыми машинами.

15) Поиск документов без прямых ссылок на сайте

Это уникальный робот, который можно отнести к категории поисковика в «Глубоком Интернете». Его задача, проанализировав имеющиеся ссылки на статические объекты (документы, презентации, картинки и т.д.), построить вероятные названия (URL) для документов, которые не предназначены для общего доступа, и проверить их наличие.

На выходе демонстрируется список из таких объектов.