**Отчет о проведенном аудите (пентесте) приложения**

# **Исходные данные**

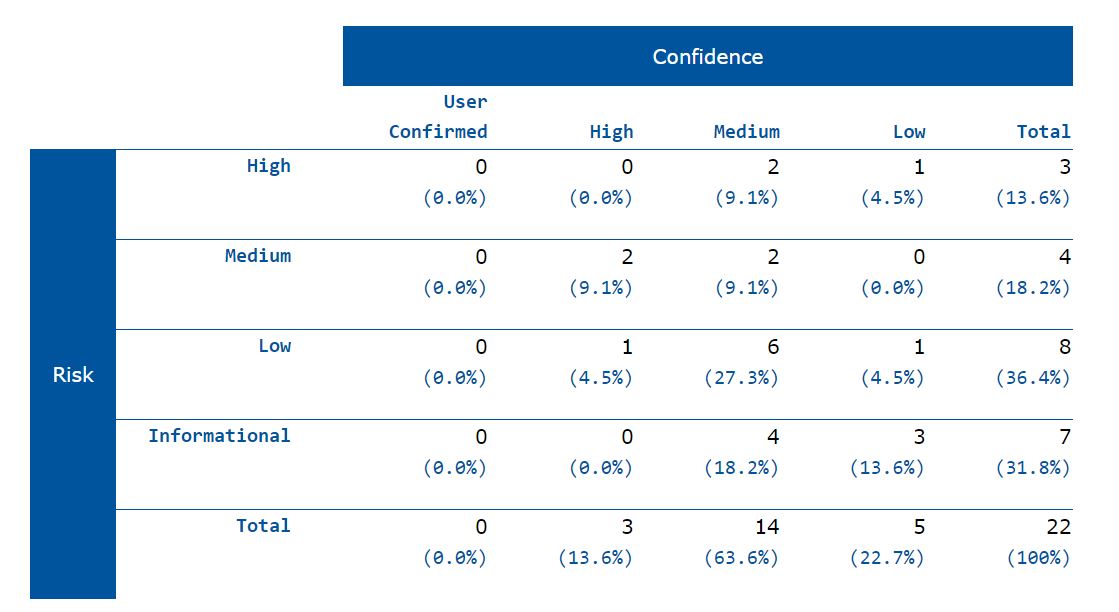
1. Предоставлен файл docker-compose.yml расположенный по адресу:  
   https://github.com/netology-code/ibdef-homeworks/blob/master/07\_pentest\_2/assets/docker-compose.yml
2. Для проведения пентеста был использован инструмент ZAP (Zed Attack Proxy)

# **Отчет по результатам аудита**

**2.1 Сайты для аудита**

На виртуальной машине был поднят docker образ и запущено приложение (имя) на <http://localhost:8080>

Суммарная таблица, в которой показано количество предупреждений для каждого уровня риска и достоверности:



## **2.2 Подробное описание четырех найденных уязвимостей**

1. **Cross Site Scripting (Persistent) и Cross Site Scripting (Reflected)**

Риск: высокий

Возможность исполнения: средняя

CWE 79

Описание:

межсайтовый скриптинг (XSS) — это метод атаки, который включает повторение предоставленного злоумышленником кода в экземпляр браузера пользователя. Экземпляром браузера может быть стандартный клиент веб-браузера или объект браузера, встроенный в программный продукт, такой как браузер в WinAmp, программа для чтения RSS или клиент электронной почты. Сам код обычно пишется на HTML/JavaScript, но также может распространяться на VBScript, ActiveX, Java, Flash или любую другую технологию, поддерживаемую браузером.

Подтверждения нахождения:

</li><scrIpt>alert(1);</scRipt><li>

1. **Content Security Policy (CSP) Header Not Set**

Риск: средний

Возможность исполнения: высокая

CWE 693

Описание:

политика безопасности содержимого (CSP) — это дополнительный уровень безопасности, который помогает обнаруживать и смягчать определенные типы атак, включая межсайтовые сценарии (XSS) и атаки с внедрением данных. Эти атаки используются для всего: от кражи данных до порчи сайта или распространения вредоносных программ. CSP предоставляет набор стандартных HTTP-заголовков, которые позволяют владельцам веб-сайтов объявлять утвержденные источники контента, которые браузеры должны разрешить загружать на эту страницу. Охватываемые типы включают JavaScript, CSS, HTML-фреймы, шрифты, изображения и встраиваемые объекты, такие как апплеты Java. ActiveX, аудио и видео файлы.

1. **Sub Resource Integrity Attribute Missing**

Риск: средний

Возможность исполнения: высокая

CWE 345

Описание:

атрибут целостности отсутствует в теге скрипта или ссылки, обслуживаемом внешним сервером. Тег целостности не позволяет злоумышленнику, получившему доступ к этому серверу, внедрить вредоносный контент.

Подтверждения нахождения:

<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">

1. **Vulnerable JS Library**

Риск: средний

Возможность исполнения: средняя

CWE 829

Описание:

Выявленная библиотека jquery версии 3.2.1 уязвима.

Подтверждения нахождения:

jquery-3.2.1.min.js

# **Рекомендации по исправлению или нейтрализации перечисленных уязвимостей**

1. **Cross Site Scripting (Persistent) и Cross Site Scripting (Reflected)**

Исправление:

для каждой созданной веб-страницы используйте и укажите кодировку символов, например, ISO-8859-1 или UTF-8.

Чтобы смягчить атаки XSS на файл cookie сеанса пользователя, установите для файла cookie сеанса значение HttpOnly. В браузерах, поддерживающих функцию HttpOnly (например, в более поздних версиях Internet Explorer и Firefox), этот атрибут может препятствовать доступу файла cookie сеанса пользователя для вредоносных клиентских сценариев, использующих document.cookie.

Предположим, что все входные данные являются вредоносными. Используйте стратегию проверки входных данных «принятие заведомо хороших», т. е. используйте разрешенный список допустимых входных данных, которые строго соответствуют спецификациям. Отклоняйте любые входные данные, которые строго не соответствуют спецификациям, или преобразуйте их во что-то, что им соответствует.

При проверке входных данных учитывайте все потенциально важные свойства, включая длину, тип входных данных, полный диапазон допустимых значений, отсутствующие или дополнительные входные данные, синтаксис, согласованность между связанными полями.

Убедитесь, что вы выполняете проверку ввода на четко определенных интерфейсах в приложении. Это поможет защитить приложение, даже если компонент повторно используется или перемещается в другое место.

1. **Content Security Policy (CSP) Header Not Set**

Исправление:

убедитесь, что ваш веб-сервер, сервер приложений, балансировщик нагрузки и т. д. настроены для установки заголовка Content-Security-Policy.

1. **Sub Resource Integrity Attribute Missing**

Исправление:

укажите допустимый атрибут целостности для тега.

1. **Vulnerable JS Library**

Исправление:

обновить jquery до последней версии.