## Выявление угроз в проектируемой системе

В соответствии с рисунком 14 в системе присутствуют следующие компоненты:

* Прокси-сервер nginx;
* Frontend-сервер Node.js;
* Backend-сервер Golang;
* Веб-сервис моделирования Golang;
* База данных PostgreSQL.

Для перечисленных компонентов будут рассмотрены возможные уязвимости.

### Нарушение доступности

Нарушение доступности может произойти вследствие атаки на ряд компонентов системы.

* вывод из строя сервера nginx, ввиду следующих причин:
  + высокая нагрузка и недостатка ресурсов;
  + воздействие DDoS атаки (формирование большого количества запросов);
  + большого количества медленных клиентов (один из вариантов DDos атаки);
  + проблема «толстых клиентов» (необходимо проксировать недопустимо большие объемы данных).
* преднамеренное большое количество запросов на получение веб-приложения, приводящее к большому времени ожидания и снижению доступности приложения.

### Нарушение целостности рассматриваемой автоматизированной системы

Нарушение целостности может быть вызвано выводом из строя одного из компонентов программного обеспечения данной программной системы.

* внедрение в веб-приложение вредоносного sql-кода и запросов;
* целенаправленное создание нагрузки на компоненты системы, обеспечивающие жизнеспособность системы;
* уничтожение физических серверов, если система не имеет резервов, неизбежно ведет к нарушению целостности.

### Нарушение конфиденциальности информации

* внедрение вредоносного SQL-кода (sql-инъекция);
* кража данных пользователей посредством XSS;
* атака CSRF для перенаправления пользователя и осуществления несанкционированных действий от его лица.

### Резюме на выявленные угрозы

Основным источником угрозы является пользователь-злоумышленник (внешняя преднамеренная угроза).

Основными объектами угрозы является аккаунты пользователя и их приватная информация.

Возможные способы несанкционированного доступа – похищение пароля пользователя, перехват сессии, внедрение вредоносного кода в компоненты программной системы.

Возможные способы устранения выявленных уязвимостей представлены в разделе 3.6.

## Управление доступом на основе ролей

Несмотря на то, что разрабатываемая система не преследует никаких коммерческих целей, что автоматически означает, что в системе не будет осуществляться товарно-денежный оборот, а также в системе не планируется хранение специфичной информации пользователя, которую он мог бы посчитать конфиденциальной, все же есть необходимость в приватности данных пользователей, а также разделение пользователей системы по ролям.

Разделение по ролям позволит наделить часть привилегированных пользователей особыми правами по модификации и просмотру данных, что необходимо в рамках администрирования системы.

Для разрабатываемой системы планируется выделить всего две роли:

* пользователь;
* администратор.

Где пользователь – обычный пользователь системы, а администратор – пользователь с особыми правами. Планируется, что администраторами будут являться сами же разработчики системы или персонал, обеспечивающий работоспособность системы.

Естественно, в отличии от администраторов, пользователи не должны обладать правами просмотра приватных данных друг друга.

## Обеспечение информационной безопасности

### Построение модели нарушителя

Т.к. в ходе выявления потенциальных угроз было выяснено, что большая часть из них относится к воздействию на систему из вне, то было принято решение построить модель для такого нарушителя.

С учётом особенностей разрабатываемой системы, предполагается, что нарушителем может быть одиночка или небольшая группа авантюристов. Модели нарушителя такие, как «организация-конкурент» и «правительственная структура» не будут рассматриваться, т.к. разрабатываемая автоматизированная система не является коммерческим продуктом, а также не имеет под собой никакой политической подоплеки.

Модели нарушителей «Одиночка» и «Группа злоумышленников» представлены в таблице 11.

На основании построенных моделей нарушителей можно предугадать основные векторы атак, на основание чего разработать методы защиты системы от попыток нарушить конфиденциальность информации или целостность самой системы.

Для устранения выявленных уязвимостей предлагается предпринять ряд мер. Защита должна быть обеспечена на всех доступных уровнях в рамках разумного. Основная часть мер должна быть применена на тех элементах программной системы, на которые в первую очередь поступает на обработку пользовательский запрос.

Таблица 11 – Модели нарушителей

|  |  |  |
| --- | --- | --- |
| **Характеристика** | **Одиночка** | **Группа злоумышленников** |
| Вычислительная мощность технических средств | Персональный компьютер | ЛВС |
| Доступ к Интернету, тип каналов доступа | Модем или выделенная линия | Использование выделенных каналов с высокой пропускной способностью |
| Финансовые возможности | Сильно ограниченные | Ограниченные |
| Уровень знаний в области IT | Невысокий или средний | Средний или высокий. Множество компетенций |
| Используемые технологии | Готовые программы, самописные скрипты, использование популярных уязвимостей | Исследование атакуемой системы, поиск уязвимостей, изготовление вредоносных программ, внедрение троянов и вирусов |
| Знание о системе защиты атакуемой системы | Отсутствие знаний об особенностях системы | Могут предпринимать значительные усилия по получения информации о работе системы на протяжения длительного временного периода |
| Преследуемые цели | Эксперимент, получение реального опыта взлома, хулиганство, особые личные мотивы | Вымогательство, внесение искажений в работу системы, похищение личных данных пользователей |
| Характер действий | Скрытый | Скрытый |
| Глубина проникновения | Первое успешное воздействие или до обнаружения | До достижения поставленных целей, до обнаружения |

### Обеспечение защиты системы

#### Обеспечение защиты от нарушения доступности и целостности системы в единой точке доступа к системе

На этапе проксирования запроса от пользователя на сервере nginx можно предпринять некоторое количество мер по защите системы.

Позволяет обеспечить частичную защиту от DDoS спама, однако из-за NAT не может быть использоваться постоянно. Для надежности защита должна обеспечиваться провайдером, т.к. он обладает большими ресурсами и знаниями о конфигурации сети Интернет для блокировки нарушителей.

Данное ограничение позволяет значительно сэкономить ресурсы в случае, если некоторая группа юзеров совершает аномально большое количество запросов, в котором нет необходимости при штатном использовании сервисов, предоставляемых веб-приложением.

Отключение медленных пользователей позволяет защититься от атак, которые используют медленное соединения для утилизации большого процента ресурсов на сервисе, на который совершается атака.

#### Обеспечение защиты в приложении frontend-сервера

Для защиты пользователя от некорректного поведения на стороне веб-приложения в браузере необходимо обеспечить защиту от наиболее частых типов атак со стороны внешнего нарушителя.

Для защиты от XSS (внедрение вредоносного кода) обеспечивается сериализация данных, которая призвана из вводимой информации удалять все типы символов, которые могут быть восприняты как исполняемый код при интерпретации браузером.

Запрос CSRF-токена – один из самых популярных и надежных на данный момент способов защиты от CSRF-атак. Обеспечивает защиту от несанкционированных действий при междоменном взаимодействии.

#### Обеспечение защиты на backend-сервере

На данном программном компоненте стоит в первую очередь обратить внимание на защиту пользовательских данных, т.к. данный компонент напрямую взаимодействует с базами данных.

Использование экранированных входных\выходных данных. Применение встроенных функций для очистки кода от вредоносных скриптов.

Для взаимодействия со всеми компонентами программной системы должен использоваться подход «белых списков», когда разрешено взаимодействие только с теми хостами, которые были явно прописаны в списке разрешенных. В этом случае какие-либо взаимодействие с «чужими хостами» будет запрещено.

Это также обеспечивает дополнительную защиту, когда проводится попытка нарушения работоспособности системы изнутри.

#### Обеспечение защиты данных в БД

Осуществляется непосредственно средствами реляционной базы данных PostgreSQL.

Для взаимодействия с базой данных требуется пройти идентификацию и аутентификацию.

Перед совершением какого-либо действия проводится авторизация пользователя.

Можно явно заблокировать для пользователей БД операции по удалению и модификации данных, без предварительного резервного копирования изменяемых данных.

Наиболее ценные данные в базе должны храниться в зашифрованном виде. Например, пароли.