**Виды уязвимостей для серверов (ФСТЭК РОССИИ)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Код** | **Дата обнаружения** | **Описание** | **Способы устранения** | **Уровень опасности (Оценка по CVSS 3.0)** |
| BDU:2020-05726 | 05.03.2020 | Уязвимость библиотеки Django для языка программирования Python связана с недостаточной защитой структуры SQL-запроса. Эксплуатация уязвимости может позволить нарушителю, действующему удалённо, выполнить произвольный код | [Метод-1](https://www.djangoproject.com/weblog/2020/mar/04/security-releases/) | 8,8 |
| BDU:2020-01767 | 02.08.2019 | Уязвимость модуля HTMLParser функции django.utils.html.strip\_tags фреймворка для веб-разработки Django связана с ошибкой медленного оценивания больших входных данных, которые содержат большие последовательности неполных HTML объектов. Эксплуатация уязвимости может позволить нарушителю, действующему удаленно, вызвать отказ в обслуживании | Для Django:  Обновление программного обеспечения до 2:2.2.4-1 или более поздней версии | 7,5 |
| BDU:2020-01459 | 15.12.2019 | Уязвимость фреймворка для веб-приложений Django связана с некорректной проверкой электронного адреса после трансформации символов Юникода в регистр, что дате возможность отправить токен сброса. Эксплуатация уязвимости может позволить нарушителю, действующему удаленно, оказать воздействие на целостность данных | Для Django:  Обновление программного обеспечения до 2:2.2.10-1 или более поздней версии | 7,5 |
| BDU:2019-01260 | 04.01.2019 | Уязвимость библиотеки Django для языка программирования Python связана с недостаточной нейтрализацией специальных элементов в выходных данных, используемых нижестоящим компонентом. Эксплуатация уязвимости может позволить нарушителю, действующему удаленно, вызвать отказ в обслуживании | Для python-django:  Обновление программного обеспечения до более поздней версии | 6,5 |
| BDU:2019-03594 | 26.07.2019 | Уязвимость функций django.contrib.postgres.fields.HStoreField и django.contrib.postgres.fields.JSONField фреймворка для веб-разработки Django связана с ошибкой преобразовании и поиска ключей, а так же поиска индексов. Эксплуатация уязвимости может позволить нарушителю, действующему удаленно, оказать воздействие на целостность данных, получить несанкционированный доступ к защищаемой информации, а также вызвать отказ в обслуживании | Для Django:  Обновление программного обеспечения до 2:2.2.4-1 или более поздней версии | 9,8 |
| BDU:2020-01759 | 02.08.2019 | Уязвимость функции jango.utils.encoding.uri\_to\_iri фреймворка для веб-разработки Django связана с неправильным кодированием последовательностей октетов UTF-8, которое может привести к чрезмерному использованию памяти из-за рекурсии. Эксплуатация уязвимости может позволить нарушителю, действующему удаленно, вызвать отказ в обслуживании | Для python-django:  Обновление программного обеспечения до более поздней версии | 7,5 |
| BDU:2020-01765 | 02.08.2019 | Уязвимость функции django.utils.text.Truncator методов chars() и words() фреймворка для веб-разработки Django связана с ошибочной передачей значения True аргументу html. Эксплуатация уязвимости может позволить нарушителю, действующему удаленно, вызвать отказ в обслуживании | Для Django:  Обновление программного обеспечения до 2:2.2.4-1 или более поздней версии | 7,5 |

**Виды угроз для серверов (ФСТЭК РОССИИ)**

| **№ п/п** | **Идентификатор** | **Описание угрозы** | **Источники угрозы** | **Объекты воздействия** | **Нарушаемые свойства безопасности информации** |
| --- | --- | --- | --- | --- | --- |
|  | УБИ.016 | Угроза заключается в возможности передачи нарушителем дискредитируемому браузеру запроса на доступ к файловой системе пользователя вместо URL-запроса. При этом браузер выполнит этот запрос с правами, которыми он был наделён при запуске, и передаст данные, полученные в результате выполнения этой операции, нарушителю.  Данная угроза обусловлена слабостями механизма проверки вводимых пользователем запросов, который не делает различий между запросами на доступ к файловой системе и URL-запросами.  Реализация данной угрозы возможна в случае наличия у нарушителя привилегий на отправку запросов браузеру, функционирующему в дискредитируемой системе | Внешний нарушитель со средним потенциалом | Сетевое программное обеспечение | К |
|  | УБИ.173 | Угроза заключается в возможности неправомерного осуществления нарушителем массовой рассылки коммерческих, политических, мошеннических и иных сообщений на веб-сервер без запроса со стороны дискредитируемых веб-серверов.  Данная угроза обусловлена уязвимостями механизмов фильтрации сообщений, поступающих из сети Интернет.  Реализация данной угрозы возможна при условии наличия в дискредитируемом веб-сервере активированного функционала, реализующего различные почтовые сервера, службы доставки мгновенных сообщений, блоги, форумы, аукционы веб-магазинов, онлайн-сервисы отправки SMS-сообщений, онлайн-сервисы голосования и др. | Внешний нарушитель с низким потенциалом | Сетевое программное обеспечение | Д |
|  | УБИ.153 | Угроза заключается в возможности осуществления нарушителем опосредованного деструктивного программного воздействия на дискредитируемую систему большим объёмом сетевого трафика, генерируемого сторонними серверами в ответ на сетевые запросы нарушителя, сформированные от имени дискредитируемой системы. Генерируемый сторонними серверами сетевой трафик значительно превышает объём сетевых запросов, формируемых нарушителем.  Данная угроза обусловлена слабостями мер межсетевого экранирования дискредитируемой информационной системы, мер контроля подлинности сетевых запросов на сторонних серверах, а также слабостями модели взаимодействия открытых систем.  Реализация данной угрозы возможна при условии наличия у нарушителя:  сведений о сторонних серверах с недостаточными мерами контроля подлинности сетевых запросов;  сведений о сетевом адресе дискредитируемой системы;  специального программного обеспечения, реализующего функции генерации сетевых пакетов | Внутренний нарушитель с низким потенциалом  Внешний нарушитель с низким потенциалом | Информационная система, сетевой узел, системное программное обеспечение, сетевое программное обеспечение | Д |

**Виды уязвимостей для базы данных PostgreSQL (ФСТЭК РОССИИ)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Код** | **Дата обнаружения** | **Описание** | **Способы устранения** | **Уровень опасности (Оценка по CVSS 3.0)** |
| BDU:2020-05467 | 12.11.2020 | Уязвимость компонента client системы управления базами данных PostgreSQL связана с использованием криптографических алгоритмов, содержащих дефекты. Эксплуатация уязвимости может позволить нарушителю, действующему удалённо, реализовать атаку типа «человек посередине» | [Метод-1](https://www.postgresql.org/support/security/) | 8,1 |
| BDU:2020-05466 | 12.11.2020 | Уязвимость реализации мета-команды «gset» системы управления базами данных PostgreSQL связана с ошибками переключения контекста привилегий. Эксплуатация уязвимости может позволить нарушителю, действующему удалённо, выполнить произвольный код | [Метод-1](https://www.postgresql.org/about/news/postgresql-131-125-1110-1015-9620-and-9524-released-2111/) | 7,5 |
| BDU:2020-05465 | 12.11.2020 | Уязвимость компонента core server системы управления базами данных PostgreSQL связана с недостаточной защитой структуры запроса SQL. Эксплуатация уязвимости может позволить нарушителю, действующему удалённо, оказать влияние на целостность, доступность и конфиденциальность данных | [Метод-1](https://www.postgresql.org/support/security/) | 8,8 |
| BDU:2020-04047 | 29.04.2020 | Уязвимость системы управления базами данных PostgreSQL связана с выходом за границы памяти. Эксплуатация уязвимости может позволить нарушителю вызвать отказ в обслуживании | Обновление программного обеспечения до 9.6.10-astrase6 или более поздней версии | 4,3 |
| BDU:2020-01320 | 20.11.2019 | Уязвимость скрипта pg\_ctlcluster пакета postgresql-common связана с небезопасным управлением привилегиями. Эксплуатация уязвимости может позволить нарушителю повысить свои привилегии | [Метод-1](https://blog.mirch.io/2019/11/15/cve-2019-3466-debian-ubuntu-pg_ctlcluster-privilege-escalation/) | 7,8 |
| BDU:2019-03221 | 08.08.2019 | Уязвимость функции SECURITY DEFINER системы управления базами данных PostgreSQL связана с непринятием мер по защите структуры запроса SQL. Эксплуатация уязвимости может позволить нарушителю, действующему удалённо, выполнить произвольные SQL команды | [Метод-1](https://git.postgresql.org/gitweb/?p=postgresql.git;a=commitdiff;h=8673743)  [Метод-2](https://git.postgresql.org/gitweb/?p=postgresql.git;a=commitdiff;h=752fa3d) | 7,5 |
| BDU:2019-02385 | 20.06.2019 | Множественные уязвимости системы управления базами данных PostgreSQL вызваны переполнением буфера на стеке. Эксплуатация уязвимостей может позволить нарушителю, действующему удалённо, выполнить произвольный код | Обновление программного обеспечения до более поздней версии | 7,5 |

**Рекомендации по развертыванию и настройке базы данных «PostgreSQL»**

**Установите PostgreSQL с помощью команд:**

*sudo apt-get update*

*sudo apt-get install postgresql postgresql-contrib*

**Стандартные настройки PostgreSQL**

СУБД PostgreSQL, установленная из официального репозитория Ubuntu, по умолчанию прослушивает localhost. Этот стандартный параметр блокирует автоматическое прослушивание открытых интерфейсов. Его можно изменить переопределением listen\_addresses в файле postgresql.conf.

Кроме того, файл pg\_hba.conf поддерживает только локальный адрес замыкания и соединения с сокетами домена Unix/Linux. Потому сервер не будет принимать соединения от внешних хостов.

*# Put your actual configuration here*

*# ----------------------------------*

*#*

*# If you want to allow non-local connections, you need to add more*

*# "host" records. In that case you will also need to make PostgreSQL*

*# listen on a non-local interface via the listen\_addresses*

*# configuration parameter, or via the -i or -h command line switches.*

*# DO NOT DISABLE!*

*# If you change this first entry you will need to make sure that the*

*# database superuser can access the database using some other method.*

*# Noninteractive access to all databases is required during automatic*

*# maintenance (custom daily cronjobs, replication, and similar tasks).*

*#*

*# Database administrative login by Unix domain socket*

*local all postgres peer*

*# TYPE DATABASE USER ADDRESS METHOD*

*# "local" is for Unix domain socket connections only*

*local all all peer*

*# IPv4 local connections:*

*host all all 127.0.0.1/32 md5*

*# IPv6 local connections:*

*host all all ::1/128 md5*

Эти параметры блокируют прослушивание публичных интерфейсов. Используя такие настройки и включив брандмауэр, вы сможете полностью защитить сервер от взаимодействия с публичными интерфейсами.

**Настройка удалённых подключений**

На этапе производства и при работе с конфиденциальными данными рекомендуется шифровать трафик PostgreSQL с помощью SSL (в дополнение к брандмауэру или виртуальной частной сети, VPN).

Можно использовать менее сложный вариант настройки безопасности сервера: для этого нужно включить брандмауэр на сервере баз данных и ограничить доступ к нему до заведомо безопасных хостов.

**1: Создание пользователя и БД**

Для работы вам понадобится тестовая БД и пользователь. Подключитесь к администратору СУБД, postgres, с помощью клиента psql. Флаг –i запускает интерактивный режим, что позволяет ввести учётные данные пользователя postgres; флаг –u позволяет указать пользователя.

*sudo -i -u postgres psql*

Теперь можно создать пользователя и пароль для него.

*CREATE USER 8host WITH PASSWORD 'password';*

**Примечание**: Замените условное имя пользователя и пароль более надёжными учётными данными.

Теперь нужно создать БД и передать все права на неё новому пользователю. С точки зрения безопасности пользователи должны иметь тот уровень доступа, который необходим им для работы, и только к необходимым ресурсам. То есть, если пользователь должен читать файлы service\_1, не нужно давать ему права на запись и исполнение файлов service\_1 и service\_2

*CREATE DATABASE 8hostdb OWNER 8host;*

Вы получите подтверждение:

*CREATE DATABASE*

Закройте оболочку СУБД:

*\q*

**2: Настройка UFW**

Для наилучшей безопасности, рекомендуется использовать встроенный брандмауэр UFW. Для этого его необходимо правильно настроить.

Проверьте состояние брандмауэра:

*sudo ufw status*

**Примечание**: Если брандмауэр отключен (inactive), включите его:

*sudo ufw enable*

После этого снова запросите состояние брандмауэра. Чтобы разблокировать SSH, введите:

*sudo ufw allow OpenSSH*

На данный момент брандмауэр поддерживает только сервис OpenSSH.

Теперь нужно разблокировать доступ к порту PostgreSQL и ограничить его только до заведомо безопасных хостов.

Приведенная ниже команда добавит правило, которое разблокирует порт PostgreSQL по умолчанию, 5432. Если вы изменили этот порт, не забудьте обновить его в следующей команде. Также вместо client\_ip\_address укажите IP-адрес клиента, которому нужен доступ к хосту. В случае необходимости повторно запустите эту команду, чтобы добавить другие IP-адреса безопасных клиентов.

*sudo ufw allow from client\_ip\_address to any port 5432*

Теперь нужно настроить PostgreSQL для прослушивания внешнего IP-адреса. Для этого нужно добавить две настройки: вход для подключения к хосту в файл pg\_hba.conf и параметр listen\_addresses в postgresql.conf.

**3: Настройка доступ к хосту**

**Добавьте настройки в файл pg\_hba.conf.**

*sudo nano /etc/postgresql/\*/main/pg\_hba.conf*

Добавьте строки host под закомментированным блоком, который описывает настройки нелокальных соединений. Также нужно добавить внешний IP-адрес сервера базы данных, чтобы быстро проверить настройку брандмауэра.

*host 8hostdb 8host client\_ip\_address/32 md5*

**4: Настройка listen\_addresses**

Теперь нужно настроить прослушивание адресов в файле postgresql.conf:

*sudo nano /etc/postgresql/9.5/main/postgresql.conf*

Найдите строку listen\_addresses и определите в ней адреса, которые нужно прослушивать. Укажите имя хоста или IP-адрес сервера БД. Убедитесь, что используете внешний IP-адрес сервера БД, а не клиента.

*#listen\_addresses = 'localhost' # what IP address(es) to listen on;*

*listen\_addresses = 'localhost,server\_ip\_address'*

Сохраните и закройте файл.

**5: Настройка логирования действий**

Данное действие позволит получить детальную информацию, в случае инцидентов.

Откройте конфигурационный файл:

*sudo vim /etc/postgresql/9.\*/main/postgresql.conf*

Допишите или раскомментируйте:

*log\_line\_prefix = '%t %c %u ' # time sessionid user*

*log\_statement = 'all'*

Управлять некоторыми параметрами можно прямо из сессии с СУБД. Например включение подробного

логирования:

*SELECT set\_config('log\_statement', 'all', true);*

По умолчанию, log-файлы хранятся по пути */var/log/postgresql/\**

**6: Перезапуск PostgreSQL**

Чтобы обновить настройки СУБД, перезапустите PostgreSQL.

*sudo systemctl restart postgresql*

Утилита systemctl не выводит результат некоторых команд. Чтобы убедиться, что сервер успешно перезапущен, запросите его состояние:

*sudo systemctl status postgresql*

Убедитесь, что сервис запущен.

**Дополнительные рекомендации**

Рекомендуется рассмотреть несколько дополнительных вариантов повышения безопасности **PostgreSQL** сервера:

* Присваивать роли пользователям для управления доступом в PostgreSQL.
* Использовать SSL-сертификат для PostgreSQL, который позволит шифровать данные.
* Использовать SSH-туннель для PostgreSQL, который защитит подключения клиентов, несовместимых с SSL.
* Использовать парольные политики в виде расширений ([1](https://github.com/postgres/postgres/tree/master/contrib/passwordcheck), [2](https://github.com/ykrysko/passwordcheck), [3](https://github.com/michaelpq/pg_plugins/tree/master/passwordcheck_extra), [4](https://github.com/devrimgunduz/passwordcheck_cracklib), [5](https://github.com/asio/password))

**Распространенные угрозы для фреймворка «Jango»**

**Межсайтовый скриптингs (XSS)**

XSS (Cross-Site Scripting — «межсайтовый скриптинг») — тип атаки на веб-системы, заключающийся во внедрении в выдаваемую веб-системой страницу вредоносного кода (который будет выполнен на компьютере пользователя при открытии им этой страницы) и взаимодействии этого кода с веб-сервером злоумышленника. Является разновидностью атаки «Внедрение кода».

Специфика подобных атак заключается в том, что вредоносный код может использовать авторизацию пользователя в веб-системе для получения к ней расширенного доступа или для получения авторизационных данных пользователя. Вредоносный код может быть вставлен в страницу как через уязвимость в веб-сервере, так и через уязвимость на компьютере пользователяs.

**Межсайтовая подделка запроса (CSRF)**

CSRF (cross-site request forgery — «межсайтовая подделка запроса», также известна как XSRF) — вид атак на посетителей веб-сайтов, использующий недостатки протокола HTTP. Если жертва заходит на сайт, созданный злоумышленником, от её лица тайно отправляется запрос на другой сервер (например, на сервер платёжной системы), осуществляющий некую вредоносную операцию (например, перевод денег на счёт злоумышленника). Для осуществления данной атаки жертва должна быть аутентифицирована на том сервере, на который отправляется запрос, и этот запрос не должен требовать какого-либо подтверждения со стороны пользователя, которое не может быть проигнорировано или подделано атакующим скриптом.

Основное применение CSRF — вынуждение выполнения каких-либо действий на уязвимом сайте от лица жертвы (изменение пароля, секретного вопроса для восстановления пароля, почты, добавление администратора и т. д.). Также с помощью CSRF возможна эксплуатация отражённых XSS, обнаруженных на другом сервере.

**SQL инъекции**

Внедрение SQL-кода (SQL injection) — один из распространённых способов взлома сайтов и программ, работающих с базами данных, основанный на внедрении в запрос произвольного SQL-кода.

Внедрение SQL, в зависимости от типа используемой СУБД и условий внедрения, может дать возможность атакующему выполнить произвольный запрос к базе данных (например, прочитать содержимое любых таблиц, удалить, изменить или добавить данные), получить возможность чтения и/или записи локальных файлов и выполнения произвольных команд на атакуемом сервере.

Атака типа внедрения SQL может быть возможна из-за некорректной обработки входных данных, используемых в SQL-запросах.

**Кликджекинг**

Кликджекинг (Clickjacking) — механизм обмана пользователей интернета, при котором злоумышленник может получить доступ к конфиденциальной информации или даже получить доступ к компьютеру пользователя, заманив его на внешне безобидную страницу или внедрив вредоносный код на безопасную страницу. Принцип основан на том, что поверх видимой страницы располагается невидимый слой, в который и загружается нужная злоумышленнику страница, при этом элемент управления (кнопка, ссылка), необходимый для осуществления требуемого действия, совмещается с видимой ссылкой или кнопкой, нажатие на которую ожидается от пользователя. Возможны различные применения технологии — от подписки на ресурс в социальной сети до кражи конфиденциальной информации и совершения покупок в интернет-магазинах за чужой счёт.

**Методы устранения распространенных угроз для фреймворка «Jango»**

**Межсайтовый скриптингs (XSS)**

Использовать систему шаблонов Django, которая защищает от большинства XSS атак, экранируя определенные символы, считающиеся "опасными" в HTML.

**Межсайтовая подделка запроса (CSRF)**

Механизм защиты заключается в добавлении тега шаблона {% csrf\_token %} в вашу форму. Этот токен будет отображен в вашем HTML как показано ниже, со значением, уникальным для каждого запрашивающего форму пользователя.

<input type='hidden' name='csrfmiddlewaretoken' value='0QRWHnYVg776y2l66mcvZqp8alrv4lb8S8lZ4ZJUWGZFA5VHrVfL2mpH29YZ39PW' />

Django генерирует уникальный для пользователя/браузера токен и отклоняет все формы, которые не содержат его или содержат его неверное значение.

**SQL инъекции**

Не использовать прямые SQL запросы в коде. Вместо этого, использовать ORM, который обеспечивает экранирование и безопасность выполнения запросов.

**Кликджекинг**

Использовать встроенную в Django защиту от кликджекинга в виде промежуточного програмного обеспечения (middleware) X-Frame-Options, который поддерживается браузерами и может запретить отображение страницы внутри <iframe>.

**Общие советы по использованию фреймворка «Jango»**

1. Wappalyzer - это расширение, которое идентифицирует программное обеспечение, работающее на нашем веб-сайте, что позволяет некоторым ботам использовать известные ошибки, прежде чем вы сможете исправить ваш веб-сайт, по этой причине рекомендуется защитить веб-сайт от этого типа расширений. В нашем случае django обнаруживается «csrfmiddlewaretoken», имя которого можно изменить и информацию о котором можно расширить, следуя последующему руководству hide-django-from-wappalyzer.

2. Одна из сильных сторон django - это панель администрирования, к которой можно получить доступ, используя URL-адрес по умолчанию \* / admin. Чтобы скрыть эту панель и избежать атак методом перебора, можно изменить этот URL-адрес. Для этого достаточно изменить файл urls.py и подставить:

# Default admin url

url(r'^admin/', admin.site.urls),

# Replace with following url

url(r'^my\_secure\_admin/', admin.site.urls),

3. Другой способ защиты нашей панели администрирования - сделать ее доступной только из сети, в которой хранится сервер, так как следует помнить, что эта панель не предназначена для конечных пользователей.

4. Обеспечить деактивацию DEBUG в производственной среде, иначе может быть предоставлен большой объем информации посетителям с плохими намерениями. Помните, что если вы оставите DEBUG равным False, вы должны настроить переменную ALLOWED\_HOSTS.

5. Использование защиты CRSF в формах, которые изменяют данные, с помощью встроенного метода промежуточного программного обеспечения CSRF «Промежуточное ПО CSRF и тег шаблона обеспечивают простую в использовании защиту от подделок межсайтовых запросов», для пользователя необходимо активировать его в настройках в качестве еще одного промежуточного программного обеспечения: 'django.middleware.csrf.CsrfViewMiddleware' и использовать тег шаблона в формах:

<form action="" method="post">{% csrf\_token %}

6. Проверять все данные, которые мы получаем в формах Django.

7. Разрешить доступ к просмотрам страниц, таким как «Логин» и «Админ», используя только протокол https.

8. При подозрении, что наш сайт подвергается атаке, можно использовать использовать такие методы, как: django-admin-honeypot, который создает нам поддельного администратора, сохранит журнал и уведомит администраторов о неудачных попытках доступа.

9. Использование шаблонов django вместо обычного html, так как это защитит нас от большинства XSS-атак.

10. Один из способов проверить имеющиеся ошибки безопасности - это использовать команду:

# With which we will obtain a detailed report about different errors and how to improve them.

python manage.py check --deploy

11. Другой способ проверить сайт Django, нет доступа к нашему серверу: ponycheckup, который предоставляет отчет об основных проблемах безопасности и способах их решения.