

# SQL injection

CZYM JEST, JAK PRZEBIEGA I JAK SIĘ PRZED TYM BRONIĆ

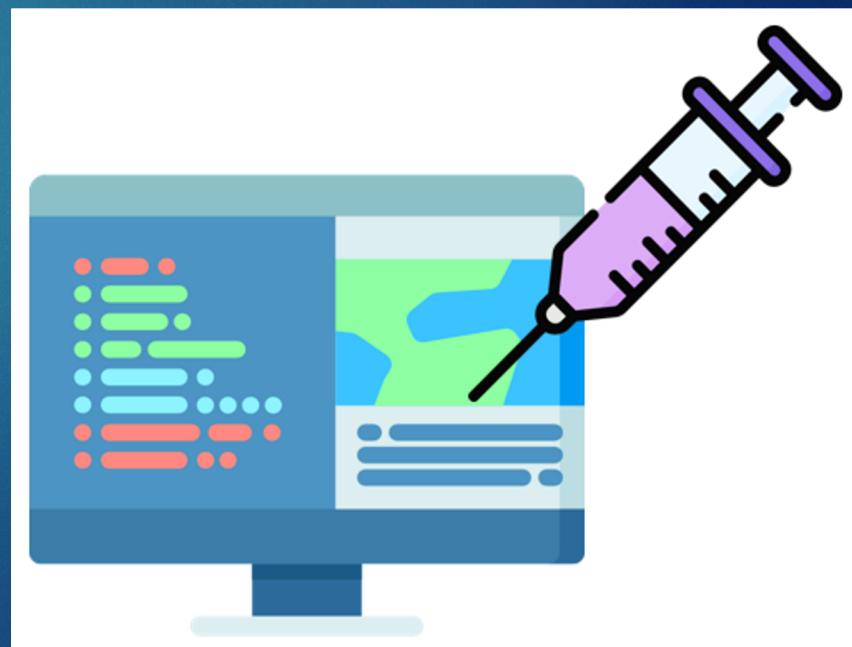
# Co to jest SQL?

- ▶ SQL(ang. Structured Query Language) – strukturalny język zapytań używany do tworzenia, modyfikowania baz danych oraz do umieszczania i pobierania danych z baz danych.
- ▶ SQL został opracowany w latach 70. w firmie IBM. Stał się standardem w komunikacji z serwerami relacyjnych baz danych.  
Potocznie mówi się, że korzystanie z relacyjnych baz danych to korzystanie z SQL.



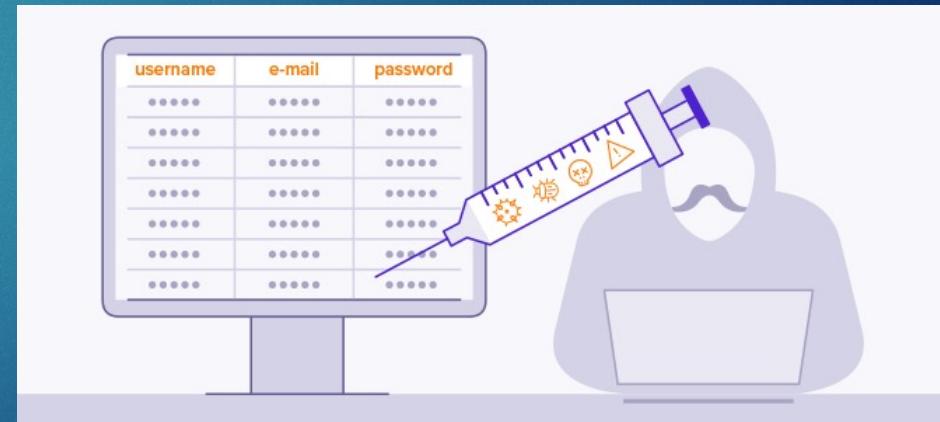
# Bezpieczeństwo

- ▶ Ponieważ SQL jest językiem interpretowanym, istnieje możliwość nadużyć w przypadku konstruowania zapytań z wykorzystaniem parametrów pochodzących z zewnętrz aplikacji.
- ▶ Szczególnie podatne na ten typ ataku są tworzone dynamicznie w oparciu o SQL-ową bazę danych serwisy internetowe. Jeśli twórca aplikacji nie zadba o sprawdzenie poprawności danych wejściowych stanowiących część zapytania, atakujący może dopisać do zapytania („wstrzyknąć”) dodatkowe komendy lub zmienić ich sposób działania. Atak taki nosi nazwę SQL injection

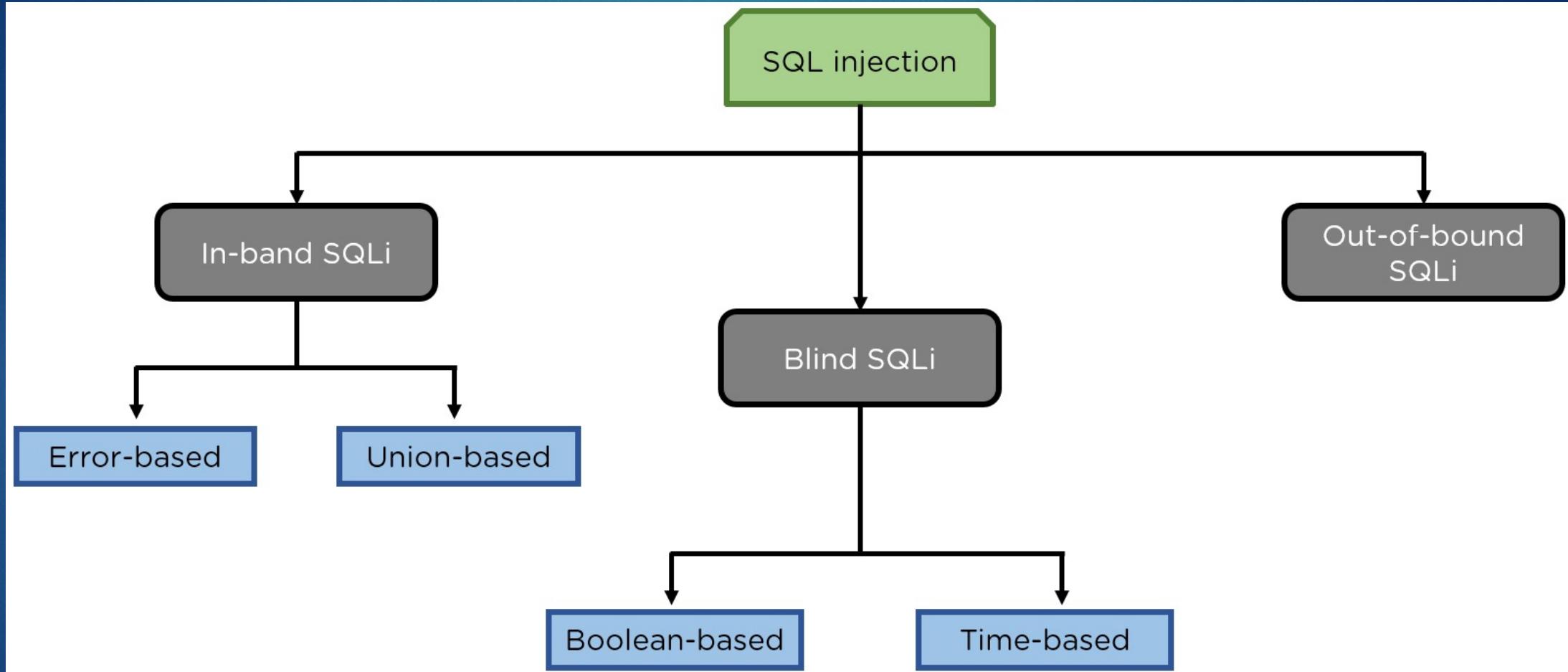


# Czym jest SQL injection?

- ▶ **SQL injection** (z ang.) – metoda ataku komputerowego wykorzystująca lukę w zabezpieczeniach aplikacji polegającą na nieodpowiednim filtrowaniu lub niedostatecznym typowaniu danych użytkownika, które to dane są później wykorzystywane przy wykonaniu zapytań (SQL) do baz danych. Podatne są na nią wszystkie systemy przyjmujące dane od użytkownika i dynamicznie generujące zapytania do bazy danych.



# Typy SQL injection



# In-band SQL injection

- ▶ In-band SQL injection – atakujący używa tego samego kanału komunikacyjnego do ataku i zebrania rezultatów.

Ten typ ataku można podzielić na:

- ▶ Error-based SQLi – atakujący działa w taki sposób, aby baza danych wygenerowała wiadomość błędu. Używając tej wiadomości atakujący może poznać jaka to baza danych, wersję serwera, etc.
- ▶ Union-based SQLi – atakujący używa operatora UNION w taki sposób, aby łącząc dwa, lub więcej zapytania SELECT otrzymać odpowiednią odpowiedź HTTP

# Out-of-bound SQL injection

- ▶ Out-of-bound SQLi nie jest popularnym typem SQLi, ponieważ zależy od funkcjonalności, które są włączone na serwerze bazy danych, który jest używany przez aplikację. Może to być np. nieprawidłowa konfiguracja ze strony administratora serwera.

# Blind SQL injection

- ▶ Blind SQLi - nie ma przesyłu danych przez aplikację webową. Atakujący nie widzi informacji zwrotnych z bazy danych. Nie zwracana jest nawet informacja o błędzie składni SQL

Blind SQL możemy podzielić na:

- ▶ Boolean-based SQLi – atakujący wysyła zapytanie SQL do bazy danych, chcąc otrzymać od aplikacji odpowiedź w zależności od tego czy zapytanie zwraca prawdę, czy fałsz
- ▶ Time-based SQLi – atakujący wysyła zapytanie SQL, które zmusza bazę danych do oczekania odpowiedniej ilości czasu, przed wysłaniem odpowiedzi. Czas odpowiedzi pomaga zidentyfikować, czy zapytanie jest prawdziwe czy nie

# Typy Blind SQL injection

- ▶ wywoływanie reakcji warunkowych
- ▶ wywoływanie błędów składniowych SQL
- ▶ wywoływanie opóźnień czasowych

# Blind SQL injection przy pomocy wywoływania reakcji warunkowych

- ▶ Założmy że mamy aplikację która zachowuje się inaczej w zależności od tego, czy zapytanie zwróci jakieś dane. Jeśli zapytanie zwróci dane wówczas na stronie wyświetlany jest komunikat powitalny
- ▶ Takie zachowanie jest wystarczające, aby móc wykorzystać lukę blind SQL injection i odzyskać informacje poprzez warunkowe wywołanie różnych odpowiedzi, w zależności od wstrzykniętego warunku.
- ▶ Przykład
  - ▶ ' AND '1'='1 – warunek prawdziwy strona zwróci komunikat powitalny
  - ▶ ' AND '1'='2 - warunek fałszywy strona nie zwróci żadnych wyników

# Przykład ataku

- ▶ Założmy, że istnieje tabela o nazwie **Users** z kolumnami **Username** i **Password** oraz użytkownikiem o nazwie **xyz**. Możemy systematycznie ustalać hasło dla tego użytkownika, wysyłając serię zapytań testujących hasło po jednym znaku na raz.
- ▶ ' AND SUBSTRING((SELECT Password FROM Users WHERE Username = 'xyz'), 1, 1) = 'a
  - ▶ Jeśli otrzymamy komunikat powitalny to znaczy że pierwszą literą hasła jest ,a'
  - ▶ Jeśli strona nie zwróci żadnych wyników to znaczy że pierwsza litera hasła jest różna od ,a'
- ▶ Zadanie 1.1

# Blind SQL injection przy pomocy wywoływania błędów składniowych SQL

- ▶ Założmy, że mamy taką samą aplikację co w poprzednim zadaniu, wykonuje ona to samo zapytanie SQL, ale nie zachowuje się inaczej w zależności od tego, czy zapytanie zwraca jakieś dane. Poprzednia technika zatem nie zadziała.
- ▶ W tej sytuacji, możliwe jest skłonienie aplikacji do zwracania warunkowych odpowiedzi poprzez warunkowe wywoływanie błędów SQL w zależności od wstrzykniętego warunku.
- ▶ Polega to na zmodyfikowaniu zapytania w taki sposób, aby powodowało ono błąd bazy danych, jeśli warunek jest prawdziwy, ale nie jeśli warunek jest fałszywy.
- ▶ Bardzo często, nieobsługiwany błąd rzucony przez bazę danych spowoduje pewną różnicę w odpowiedzi, pozwalając nam wywnioskować prawdziwość wstrzykniętego warunku.

# Przykład ataku

- ▶ ' AND (SELECT CASE WHEN (1=2) THEN 1/0 ELSE 'a' END)='a
- ▶ ' AND (SELECT CASE WHEN (1=1) THEN 1/0 ELSE 'a' END)='a
  
- ▶ Wykorzystujemy CASE do testowania warunku i zwracania innego wyrażenia w zależności od tego, czy wyrażenie jest prawdziwe.
- ▶ W pierwszym przykładzie wyrażenie CASE zwraca wartość 'a', co nie powoduje żadnego błędu.
- ▶ W drugim przypadku oblicza wartość 1/0, co powoduje błąd dzielenia przez zero. Zakładając, że błąd powoduje jakąś różnicę w odpowiedzi aplikacji, możemy użyć tej różnicy do wywnioskowania, czy wstrzyknięty warunek jest prawdziwy.

# Przykład ataku

- ▶ Używając tej techniki, możemy pobierać dane w sposób już opisany, testując systematycznie jeden znak na raz:
- ▶ ' AND (SELECT CASE WHEN (Username = 'Administrator' AND SUBSTRING>Password, 1, 1) > 'm') THEN 1/0 ELSE 'a' END FROM Users)= 'a'
- ▶ Zadanie 2.1

# Blind SQL injection przy pomocy wywoływania opóźnień czasowych

- ▶ Zapytania SQL są zazwyczaj przetwarzane synchronicznie przez aplikację, opóźnienie wykonania zapytania SQL spowoduje również opóźnienie odpowiedzi HTTP. Dzięki temu możemy wnioskować o prawdziwości wstrzykniętego warunku na podstawie czasu, jaki upłynie do otrzymania odpowiedzi HTTP.
- ▶ Techniki wyzwalania opóźnienia czasowego są bardzo specyficzne dla typu używanej bazy danych. W przypadku Microsoft SQL Server do testowania warunku i wyzwalania opóźnienia w zależności od tego, czy wyrażenie jest prawdziwe, można użyć danych wejściowych takich jak poniższe:
  - ▶ '; IF (1=2) WAITFOR DELAY '0:0:10'--.
  - ▶ '; IF (1=1) WAITFOR DELAY '0:0:10'--

# Blind SQL injection przy pomocy wywoływania opóźnień czasowych

- ▶ Używając tej techniki, możemy pobierać dane w sposób już opisany, testując systematycznie jeden znak na raz:
- ▶ `' IF (SELECT COUNT.Username) FROM Users WHERE Username = 'Administrator' AND SUBSTRING>Password, 1, 1) > ,x') = 1 WAITFOR DELAY '0:0:{delay}'—`
  
- ▶ Zadanie 3.1
- ▶ Zadanie 3.2

# Sposoby obrony przed SQL injection

- ▶ Używanie silnie spreparowanych instrukcji
- ▶ Odpowiednie procedury przechowywania danych
- ▶ Walidacja danych wejściowych z wykorzystaniem białej listy dozwolonych wyrażeń
- ▶ Filtrowanie i stosowanie znaków ucieczki w ciągach wprowadzanych przez użytkownika

# Dziękujemy za uwagę!

SZYMON OCHMAN  
NORBERT ROCZNIAK

źródła:

- [https://www.simplilearn.com/tutorials/cyber-security-tutorial/what-is-sql-injection#types\\_of\\_sql\\_injection](https://www.simplilearn.com/tutorials/cyber-security-tutorial/what-is-sql-injection#types_of_sql_injection)
- <https://pl.wikipedia.org/wiki/SQL>
- <https://www.avast.com/c-sql-injection#gref>
- <https://portswigger.net/web-security/sql-injection/blind>