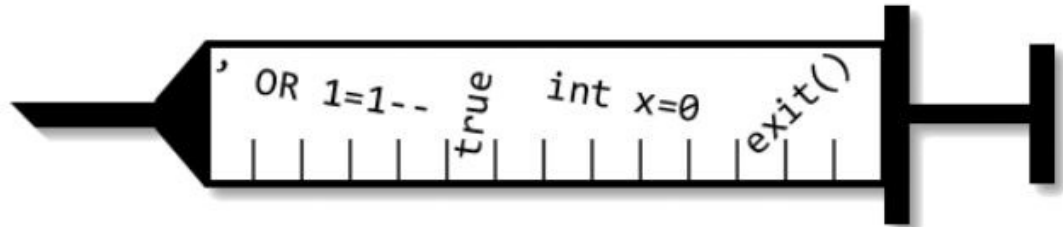


SQL-ін'єкція

Атаки та
пом'якшення
наслідків

Ін'єкція SQL

- Різновид ін'єкційних атак
- Поширена атака на веб-додатки
- Зловмисник змушує програму прийняти шкідливий код
- Викликано поганим програмуванням



Початок ін'єкції SQL

----[Conclusion

Well, that about wraps it up for now. What are the morals to the above stories?

- Don't use sample files/applications on public/production servers.
- Don't use 'local-host only' security, especially on proxys.
- Watch what exactly is changed when you upgrade.
- Don't assume user's input is ok for SQL queries.

In short, use your brain. Till next time, have fun.

rain.forest.puppy / [WT]



rfpuppy@iname.com

----[EOF

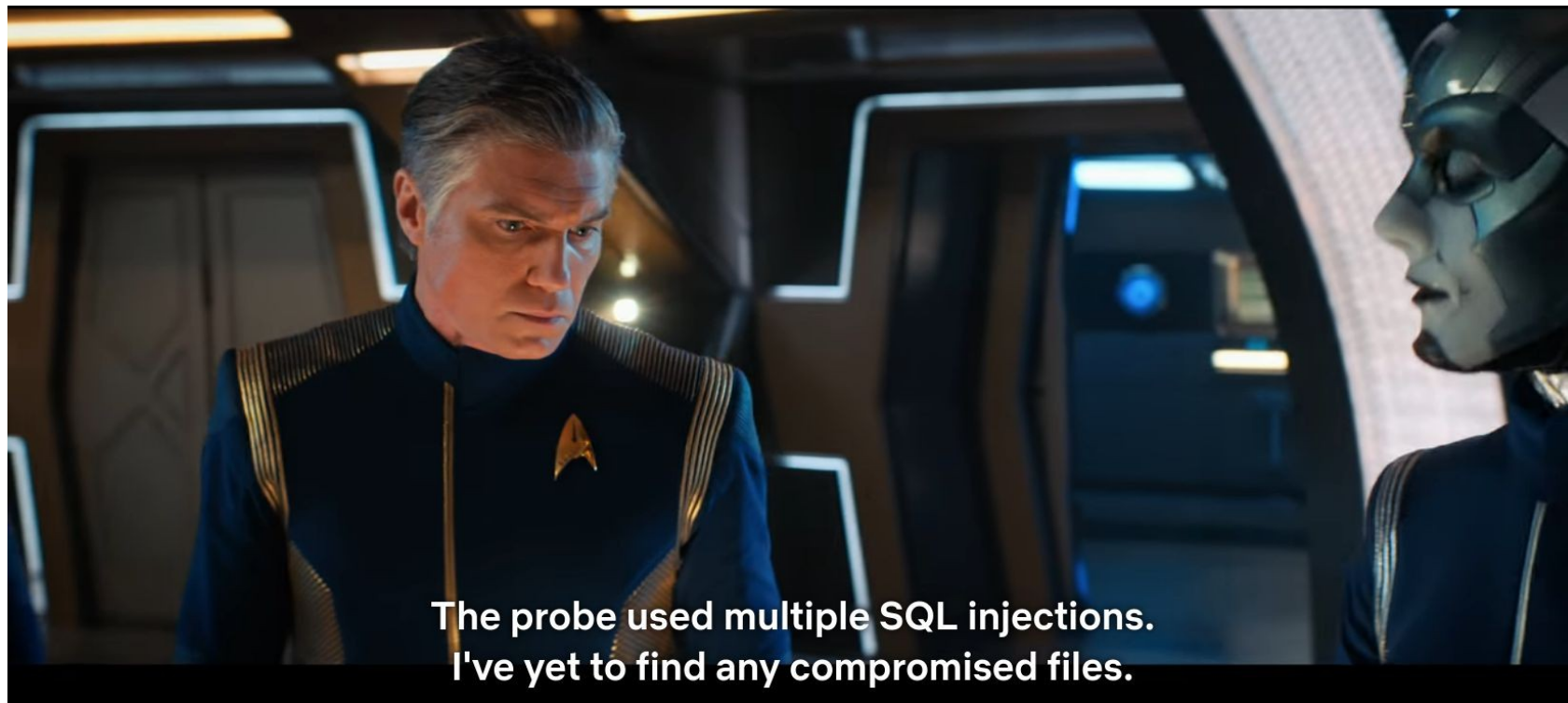
Ін'єкція SQL та OWASP



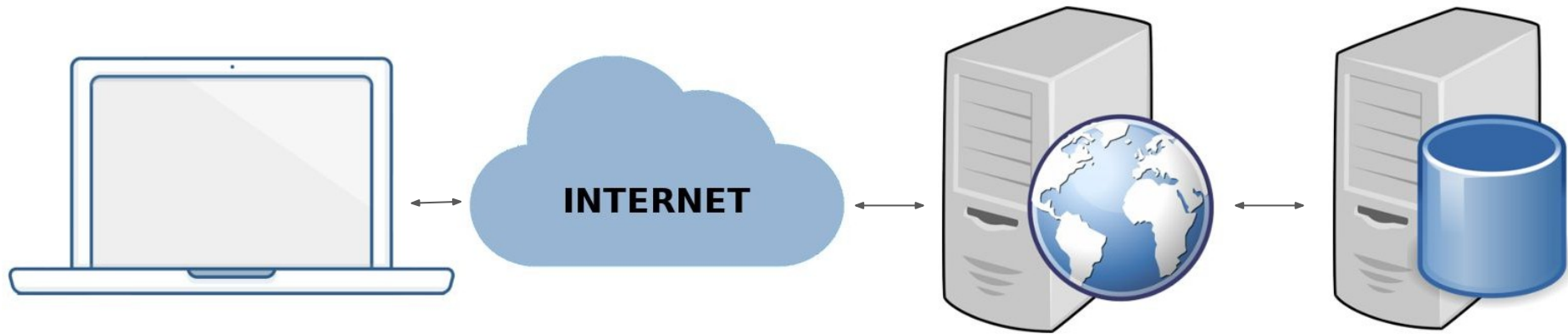
Топ-10 ін'єкцій SQL та OWASP

OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection 	→	A1:2017-Injection 
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	⊗	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	⊗	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

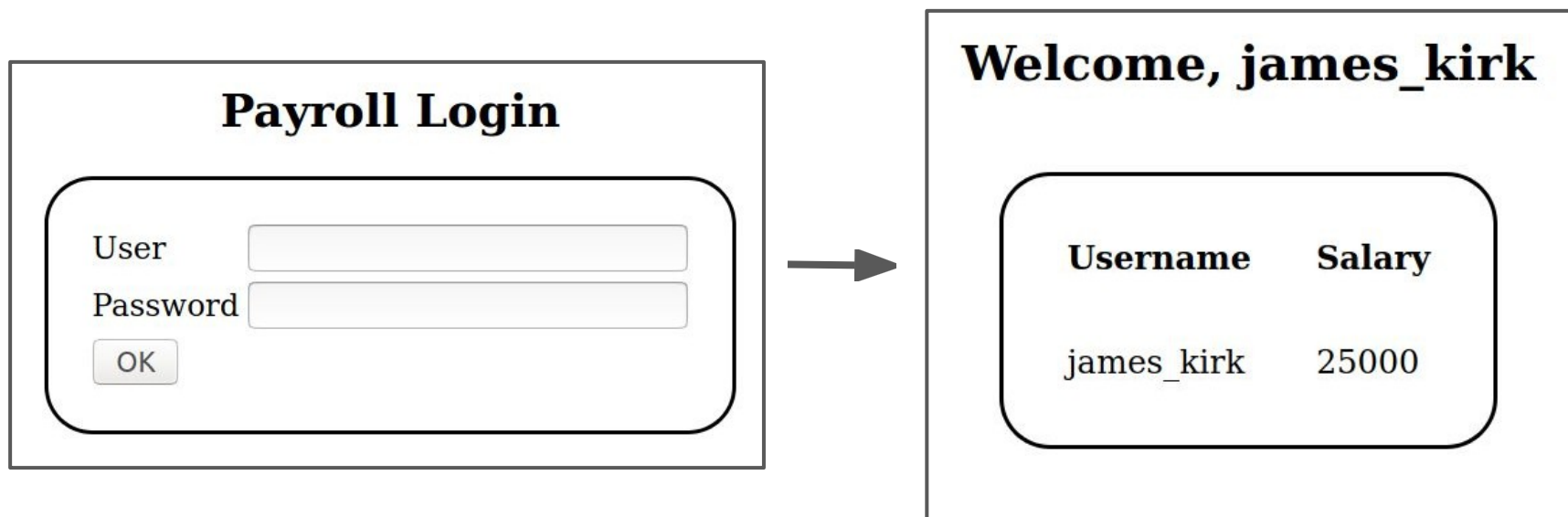
SQL-ін'єкції... все ще проблема в 23 столітті



Ін'єкції SQL та веб-додатки



Приклад веб-додатку: Фронтенд



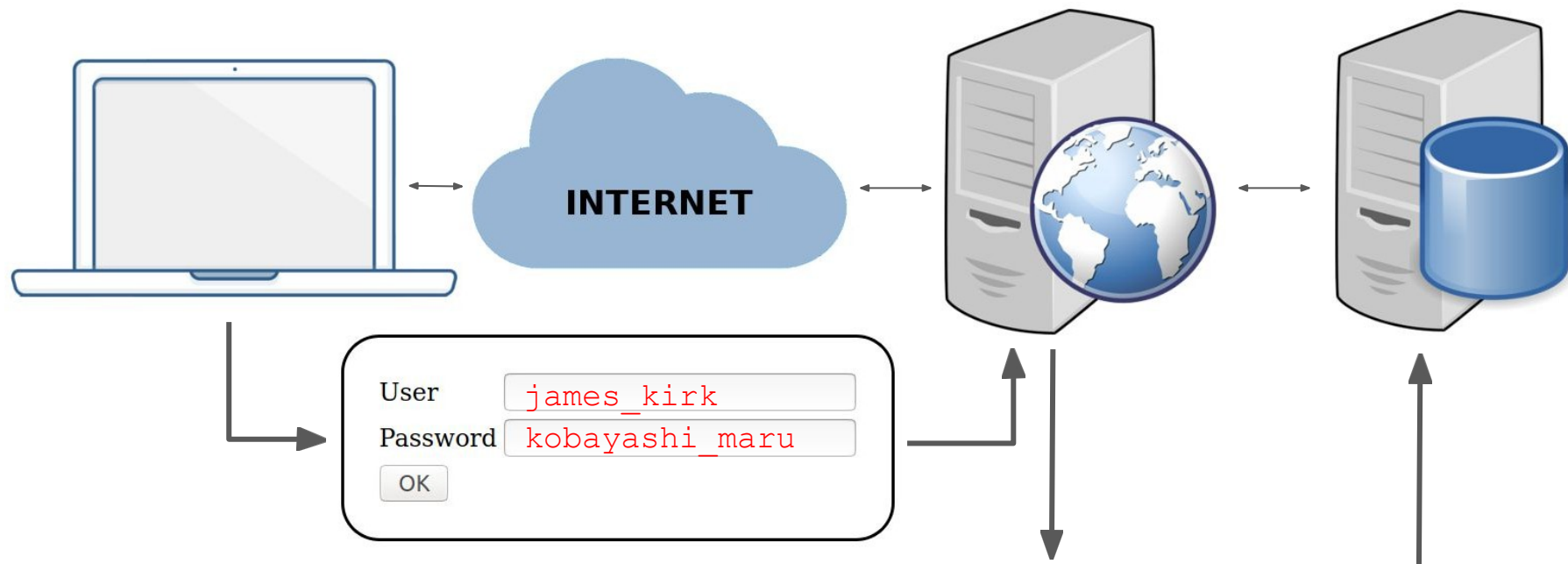
Приклад веб-додатку: Бекенд

```
mysql> select * from users;
```

username	first_name	last_name	password	salary
james_kirk	James	Kirk	kobayashi_maru	25000
mr_spock	Mr	Spock	OnlyL0g!c	99000
leonard_mccoy	Leonard	McCoy	hesDEADjim!	45000
nyota_uhura	Nyota	Uhura	StarShine	39000
montgomery_scott	Montgomery	Scott	ScottyDoesntKnow	1250
hiraku_sulu	Hikaru	Sulu	parking-break-on	3500
pavel_chekov	Pavel	Chekov	99victorvictor2	2500

```
7 rows in set (0.00 sec)
```

Звичайний запит до веб-додатку



```
SELECT username, salary FROM users  
WHERE username = 'james_kirk' AND password = 'kobayashi_maru'
```

SQL-ін'єкція

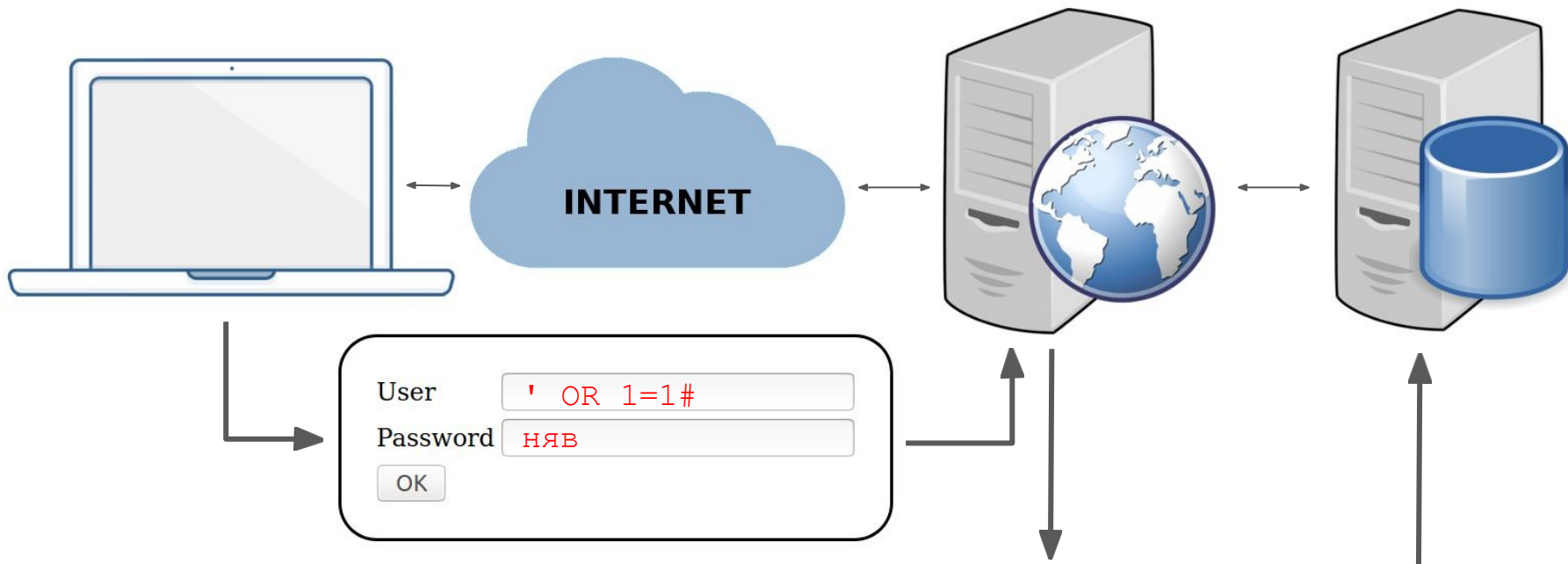
уразливість

```
# Отримати ім'я користувача/пароль від користувача
$user = $_POST['user'];
$pass = $_POST['password'];
```

```
# Скласти SQL-запит
$sql = "SELECT ім'я          зарплата   користувачі
користувача,                від
        WHERE username = '$user' AND password = '$pass'";
        ім'я користувача
```

```
# Виконати SQL-запит
$conn->multi_query($sql)
```

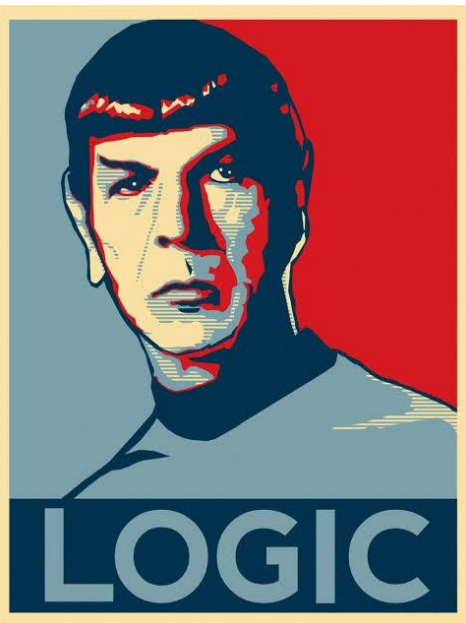
Атака на SQL ін'єкції



```
SELECT username, salary FROM users  
WHERE username = ''' OR 1=1#' AND password = 'meow'
```

Логіка атаки SQL Injection

```
SELECT username, salary FROM users  
WHERE username = '' OR 1=1# AND password = 'meow'
```



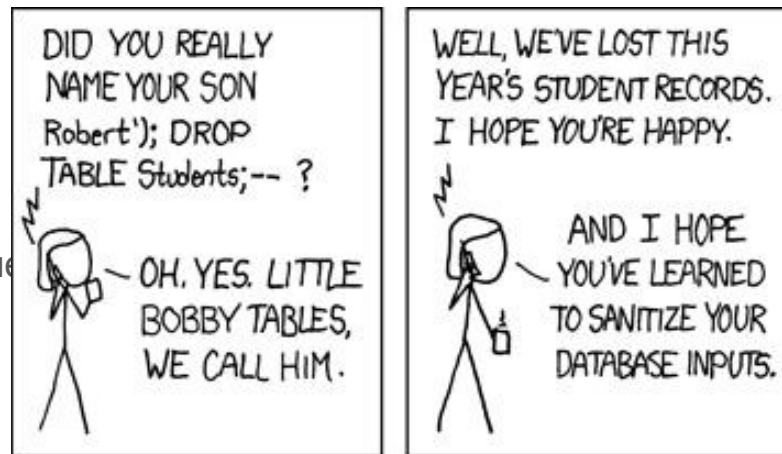
username = ''	Неправда.
1=1	Правда.

username = '' OR 1=1	Правда.
#	Коментар PHP
' AND password = 'meow'	Не бігти.

Демонстрація: Атака

Захист від SQL-ін'єкцій

- OWASP надає [шпаргалку](#) для [запобігання ін'єкції SQL](#)
- Первинний захист:
 - Використання підготовленої звітності
 - Використання збережених процедур
 - Перевірка входних даних у білому списку
 - Екранування всіх даних, введених користувачем



- Основна передумова: **ДЕЗАНАЛІЗУЙТЕ ВВЕДЕННЯ КОРИСТУВАЧА!**

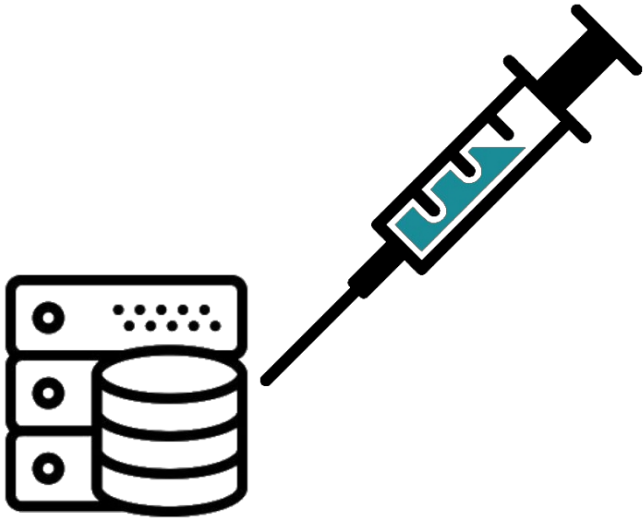
Демонстрація: Пом'якшення

Запобігання SQL-ін'єкціям:

Приклад

```
# Складання та виконання SQL-запиту
$stmt = $pdo->prepare("SELECT username, salary FROM users
                        WHERE ім'я користувача = ? і пароль =
                        ?");
$stmt->execute([$user, $pass]);
```

```
SELECT username, salary FROM users
WHERE username = ' ' OR 1=1# ' AND password = 'meow'
```



Дякую!

Питання?