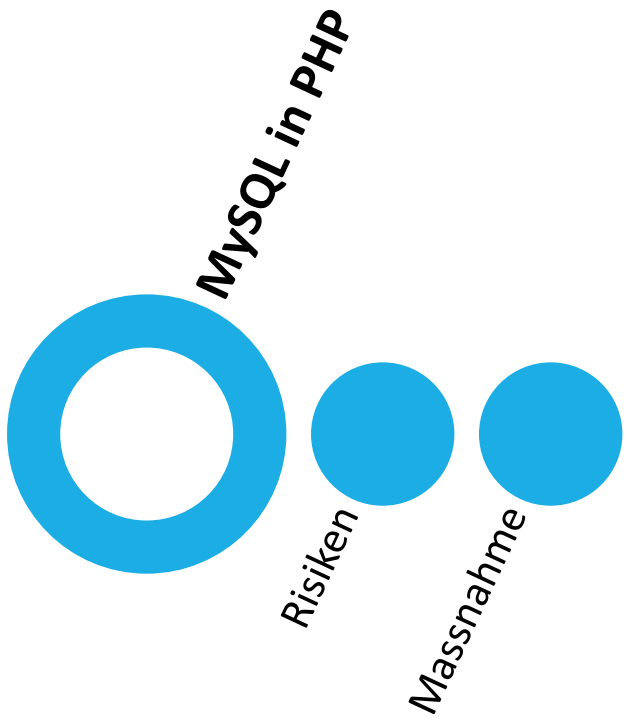
Programm diese Woche

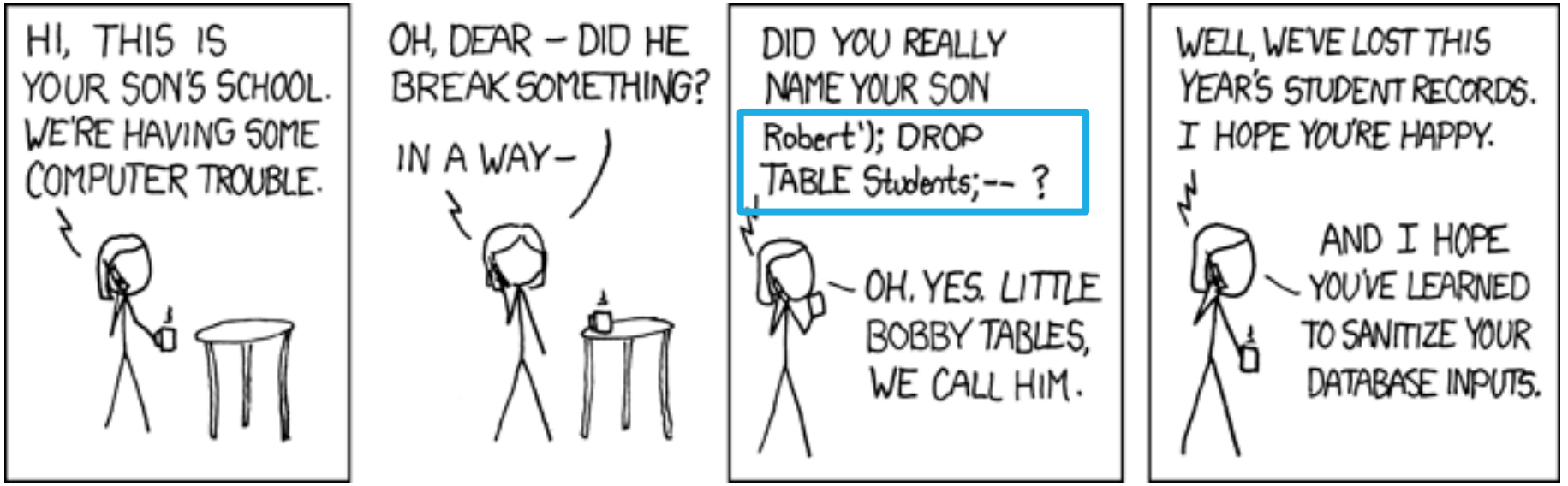


Angriffsflächen

WAS EIN HACKER VERSUCHEN WIRD…

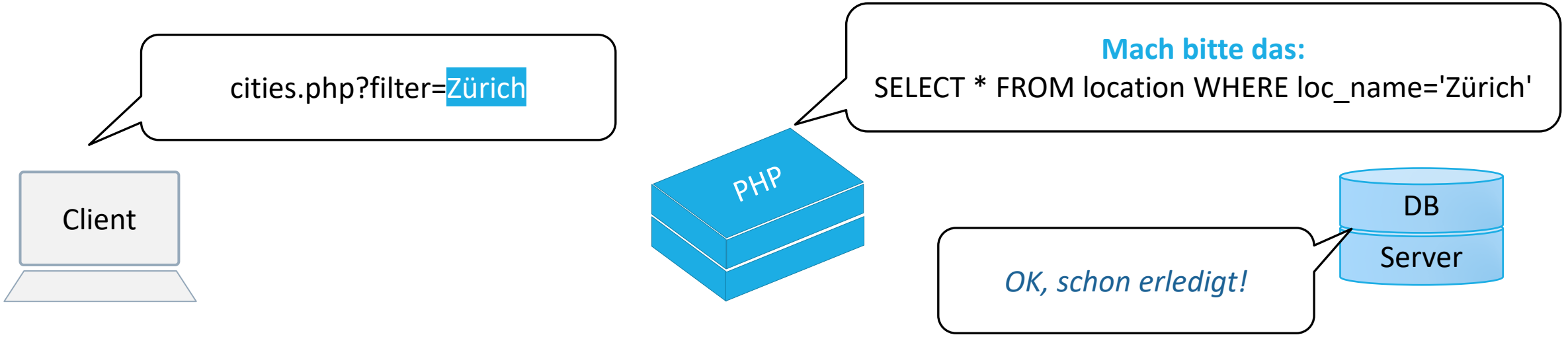
SQL-Injection: Risiko #1 in Datenbank-Applikationen

WIE PHP ALS «AUTOR» DES SQL STATEMENTS AUSGENUTZT WIRD


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

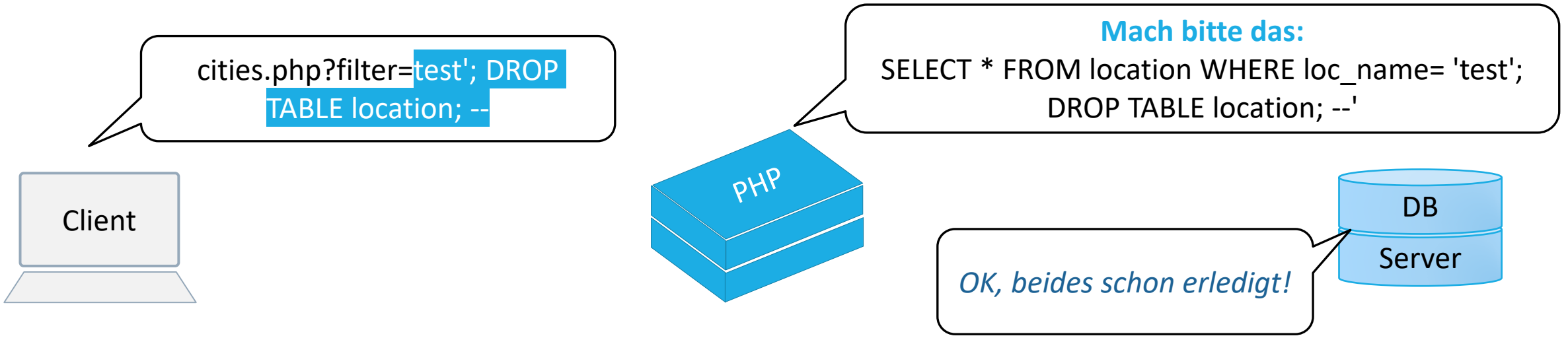
Normales SQL Statement

Kommunikation im Team Web-Technologien



Normales SQL Statement

Kommunikation im Team Web-Technologien



Problem:
- User Input könnte SQL Code enthalten

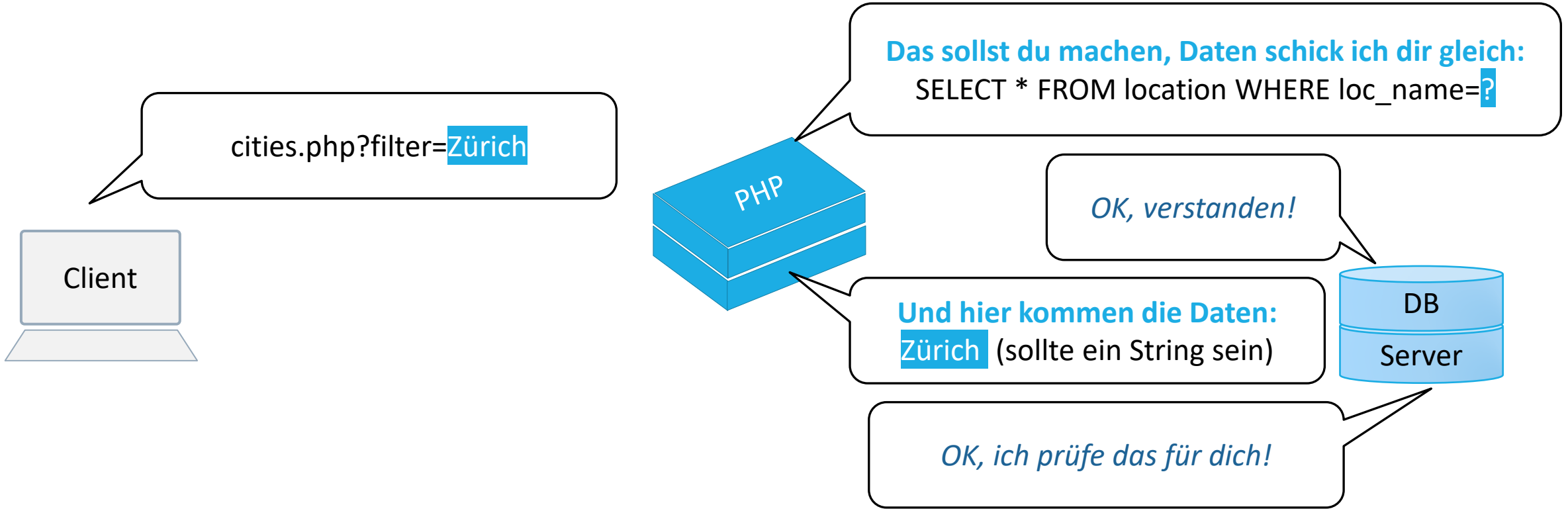
Problem:
- Validierung nur hier möglich
- Aber: PHP versteht kein SQL

Problem:
Keine Möglichkeit, User Input zu erkennen (oder zu validieren)


Ich wars nicht…
Client
PHP
DB
Server

Vorbereitetes SQL Statement

Intelligente Kommunikation im Team Web-Technologien



LÖSUNG:

TRENNUNG VON DATEN & LOGIK bis zum MySQL Server aufrecht erhalten.

Massnahmen

Die wichtigsten Massnahmen zum Schutz der SQL Datenbank

- User Input bereinigen, wo möglich – Type INT erzwingen
- Prepared Statements anwenden – **IMMER WENN USER INPUT VORHANDEN**

