software studio

injection attacks

Daniel Jackson

what is injection?

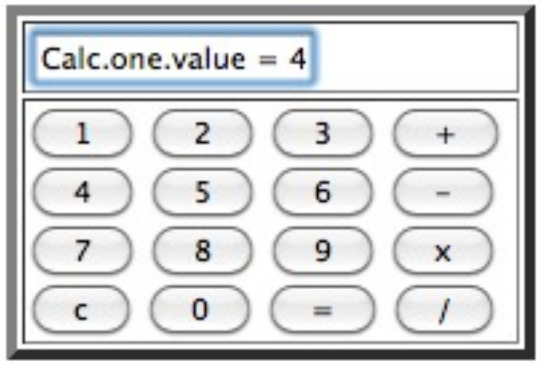
a JavaScript injection

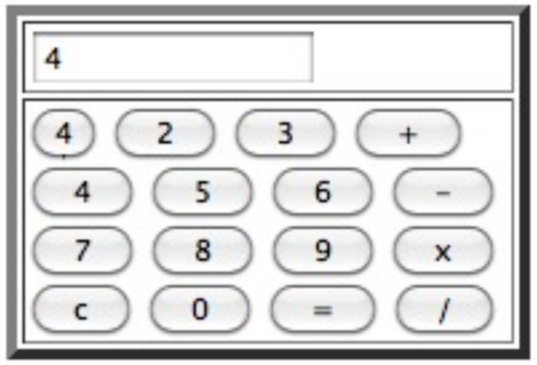
lethal combination

› strings everywhere

› eval command

```
...
<FORM NAME="Calc">
<INPUT TYPE="text" NAME="Input" Size="16">
<INPUT TYPE="button" NAME="one" VALUE="  1  "
OnClick="Calc.Input.value += '1'">
<INPUT TYPE="button" NAME="three" VALUE="3"
OnClick="Calc.Input.value += '3'">
...
<INPUT TYPE="button" NAME="plus" VALUE="+"
OnClick="Calc.Input.value += '+'">
<INPUT TYPE="button" NAME="DoIt" VALUE="="
OnClick="Calc.Input.value = eval(Calc.Input.value)">
</FORM>
```





A Javascript/HTML calculator

from http://www.javascriptkit.com/script/cut18.shtml

what is injection?

interpreters

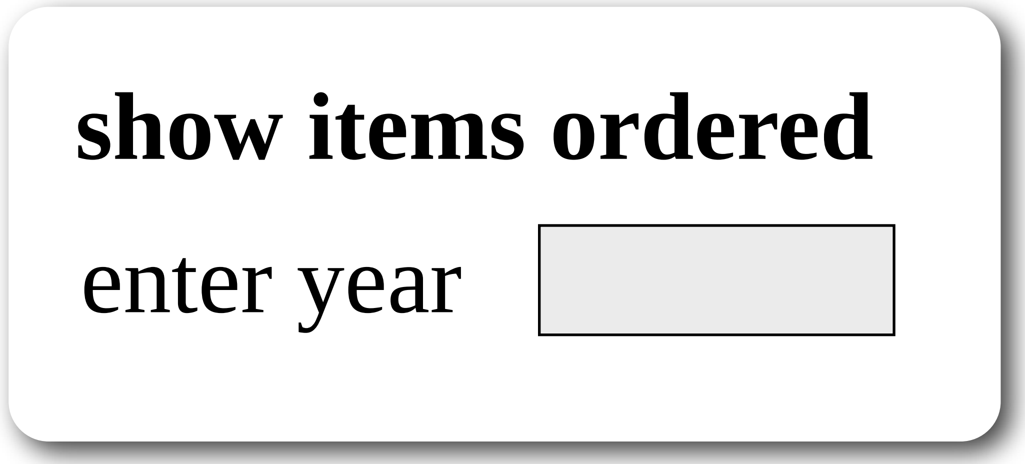
› eg, eval (JavaScript), execute (SQL)
› turn data into code
› very useful, very dangerous

JavaScript injection

› in itself, no big deal (unless JS runs server side)
› but can lead to XSS and CSRF

SQL injection

a SQL injection attack



show items ordered

enter year

```
query = “SELECT date, item FROM orders WHERE user=”
          + session[‘user_id’]
          + “AND year=” + request.form[‘year’]
execute(query)
```

an injection attack

suppose user makes a modified HTTP request

- https://www.store.com/orders?year=0%20OR%201%3D1

SELECT date, item FROM orders
WHERE user=126 AND year=0 ***OR 1=1***

effect

- sets year variable to 0 OR 1=1
- shows all orders in the database

worse

user generates this query:

```
SELECT date, item FROM orders
WHERE user=126 AND year=0
UNION
SELECT cardholder, number, exp_date FROM creditcards
```

reveals credit card database!

even worse

user generates this query:

```
SELECT date, item FROM orders
WHERE user=126 AND year=0
; DROP TABLE creditcards
```

a denial of service attack

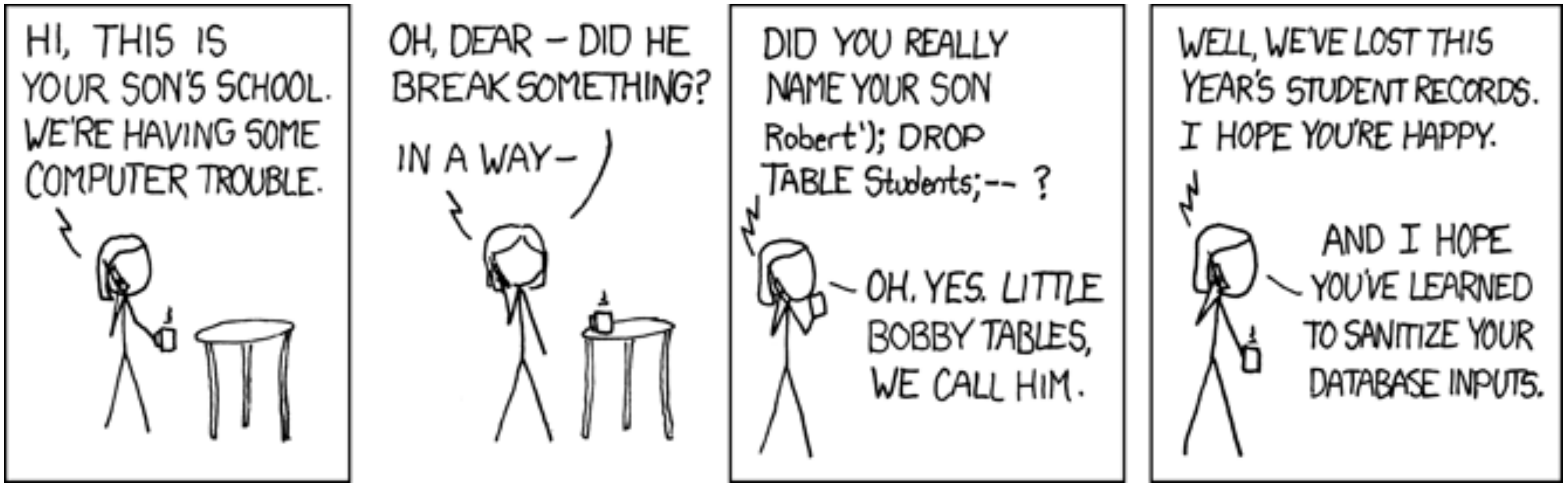
and even worse...

user generates this query

```
SELECT date, item FROM orders
WHERE user=126 AND year=0
; INSERT INTO admin VALUES (‘hacker’, ...)
```

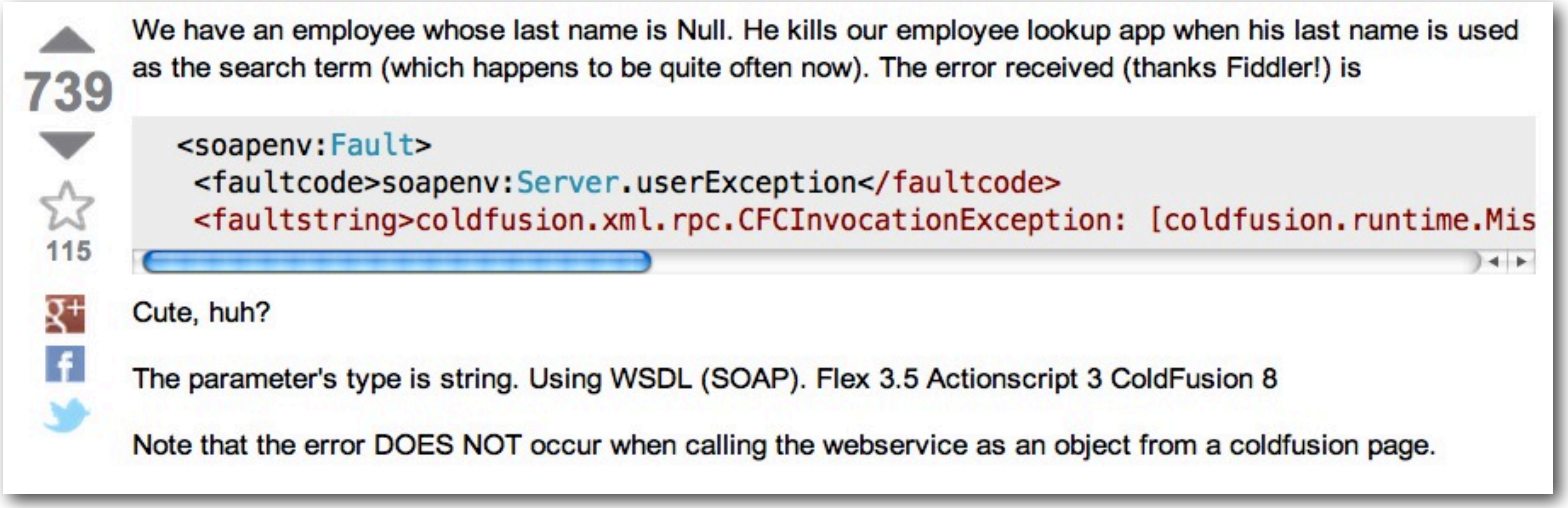
user takes over machine!

Bobby Tables


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.

from http://xkcd.com/327/

a real Bobby Tables?



739

115

We have an employee whose last name is Null. He kills our employee lookup app when his last name is used as the search term (which happens to be quite often now). The error received (thanks Fiddler!) is

```
<soapenv:Fault>
 <faultcode>soapenv:Server.userException</faultcode>
 <faultstring>coldfusion.xml.rpc.CFCInvocationException: [coldfusion.runtime.Mis
```

Cute, huh?

The parameter's type is string. Using WSDL (SOAP). Flex 3.5 Actionscript 3 ColdFusion 8

Note that the error DOES NOT occur when calling the webservice as an object from a coldfusion page.

Original question asked by bill on Stack Overflow.

shell injection

secure voting site?

Quotation removed due to copyright restrictions.
Reference: DeBonis, Mike. "Hacker Infiltration Ends D.C. Online Voting Trial," *The Washington Post*, October 4, 2010.

uploading completed PDF ballot

Screenshot of PDF ballot upload removed due to copyright restrictions.
Reference: Fig. 2f in Wolcheck, Scott, Eric Wustrow, Dawn Isabel, et al. "Attacking the Washington D.C. Internet Voting System." *Proc. 16th Conference on Financial Cryptography & Data Security* (Feb. 2012).

shell injection vulnerability

uploaded ballot saved like this:

aagh!

```
run ("gpg" , "––trust–model always –o
\"#{File.expand_path(dst.path)}\" –e –r \"#{@recipient}\"
  \"#{File .expand_path(src.path)}\"")
```

so attacker uploaded file with name

› myfile.$(command)

Unix command substitution: **execute command and replace expr by result**

see Wolchok et al. Attacking the Washington, D.C. Internet Voting System
https://jhalderm.com/pub/papers/dcvoting-fc12.pdf

even got control of camera!

see Wolchok et al. Attacking the Washington, D.C. Internet Voting System
https://jhalderm.com/pub/papers/dcvoting-fc12.pdf

preventing injection attacks

best strategy

› never call an interpreter!

if you must make commands on the fly

› build them with expression objects, not strings

for database injections

› use an ORM: no SQL queries

› use parameterized queries

bad:

```
Client.where("city = #{params[:city]}")
```

better:

```
Client.where("city = ?", params[:city])
```