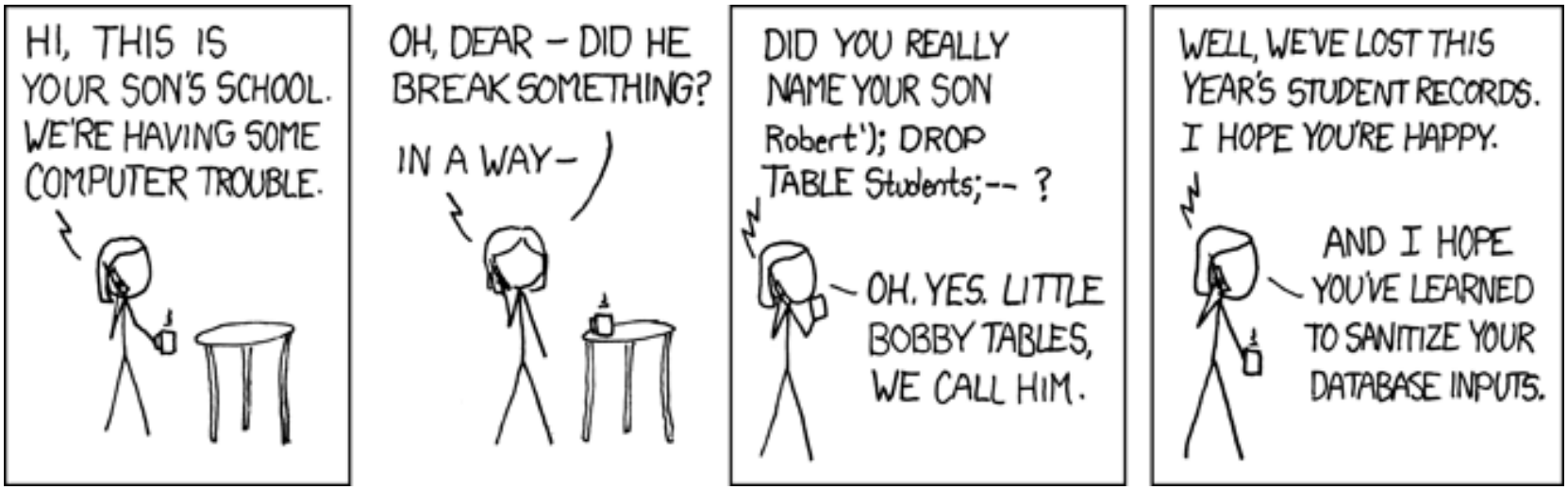
Security

Class 39


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.

Web Attacks

servers are under constant attack

borax log example

Attacker Goals

main reasons for website attacks:

access private data usernames, passwords, email addresses, social security numbers, grades, prices, PINs

change data most of the above

spoof pretend to be someone else or some other site

damage a site so it cannot be used by others; damage the site's reputation

spread malware malicious software such as viruses, worms, spyware, ransomware, adware, scareware, trojan horses, etc.

Techniques 1

main techniques used for attacks:

brute force attempting to guess passwords, filenames, etc., by trying many possible combinations until a correct one is discovered

cross site request forgery (CSRF, one-click attack) leading a browser to make a request from one malicious page to another site that performs malicious actions

cross site scripting (XSS) inserting malicious JavaScript (or HTML) into a web page to be executed

denial of service (DoS) making a website unavailable to others

exploits taking advantage of known security problems in the web server

error handling intentionally generating errors messages hoping to harvest hints for further attacks

information leakage allowing an attacker to look at data or files that give hints for further attacks

Techniques 2

malicious file execution allowing a file to be uploaded, and then executed as code or HTML, that contains malicious content

man in the middle placing a malicious machine in the network chain between client and server, to intercept, examine, and change network traffic

physical access compromising data or code by direct physical action

privilege escalation causing code to run in a privileged context, such as the root user on a Unix system or administrator on a Windows system

session hijack obtaining data from another user's cookies in order to masquerade as that user

SQL injection inserting malicious SQL code into a web site

Techniques 3

and the biggest one of all:

Techniques 3

and the biggest one of all:

social engineering tricking a user into compromising security via trust (e.g., pretexting, phishing, baiting, quid-pro-quo, tailgating)

Response

Whew! What to do?

responses fundamentally fall into two categories

1. prevention
2. detection and remediation

Response

Whew! What to do?

responses fundamentally fall into two categories

1. prevention
2. detection and remediation

- we'll talk a bit today about prevention
- for detection take CS455; much is domain of sysadmin

URL Resource Discovery

- you buy a movie on Amazon prime and you see this URL: `http://video.amazon.com/movies/2010/the-dark-knight?prepaid=true&price=399`
- think like an attacker
- what do you try?

URL Resource Discovery

- what if you try to access a site and you get redirected to: `http://www.foo.bar/login-fail.php`
- what do you do?

URL Resource Discovery

what's the difference between these two:

`http://borax.truman.edu/315/foo.html`
404: Not Found

`http://borax.truman.edu/315/bar.html`
403: Forbidden

File Resource Discovery

information leakage example

reveal.php (view page source)

Error Messages

dberror.php

XSS

- one form of code injection
- inject client-side code into a web page
- another user inadvertently views or runs the code
- based on unwarranted trust

example home.php

Stored Information

- passwords: never store in plain text
- credit card numbers: encrypt with key
- social security numbers: store separate from name

Defending Against XSS

Never Trust User Input

- never use input data in a sql query without a prepared statement
- never write input data to html without htmlspecialcharacters or htmlentities
- never use input data in a php statement without preg_match

Places to Watch For

⟨!-- HERE --⟩ never allow input data in an HTML comment

⟨tag HERE="foo"⟩ never allow input data as an attribute name

⟨HERE foo="bar"⟩ never allow input data to be a tag name

⟨style⟩ HERE ⟨/style⟩ never allow input data to be CSS code

MOST important:

⟨script⟩ HERE ⟨/script⟩ never accept actual JavaScript code from user input, ever!