Security and Software Engineering

High-Level Lecture Today!

Question 19 of 22

Select the correct answer.

The IDS monitors and collects network system information and analyzes it to detect attacks or intrusions.

- True
- I don't know

One-Slide Summary

- **Physical security** and **operating system security** are of critical importance and must be understood.
- Key issues in security, including **buffer overruns**, **virus detection**, **spam filtering**, **SQL code-injection attacks**, and **cross-site scripting** all involve cost/benefit deployment tradeoffs relevant to project management.

The State of Legacy Programming

- **Buffer overruns** are common
  - Programmers must do their own bounds checking
  - Easy to forget or be off-by-one or more
  - Program still appears to work correctly

- In C, with respect to buffer overruns, it is
  - Easy to do the wrong thing
  - Hard to do the right thing
- Java, C#, Rust, etc., all avoid this, but …

The State of Hacking

- Buffer overruns remain an attack of choice
  - 40-50% of new vulnerabilities are buffer overruns
  - Many recent attacks of this flavor: Code Red, Nimda, MS-SQL server, yada yada
  - "Buffer overflows have been the most common form of security vulnerability for the past ten years …" [OGI DARPA 2000]
  - From 2007 on, XSS and SQL-CIV are more popular, and buffer overruns are now #2
- Highly automated toolkits are available to exploit known buffer overruns
  - Look up "script kiddie"

The Sad Reality

- Even well-known buffer overruns are still widely exploited
  - Hard to get people to upgrade millions of vulnerable machines
  - Recall Equifax patch deployment timeline
- We assume that there are many more unknown buffer overrun vulnerabilities
  - At least unknown to white hats

Static Analysis to Detect Buffer Overruns

- Detecting buffer overruns *before* distributing code would be better
- Idea: Build a static analysis tool to detect buffer overruns
- This is a popular research area; we'll present one idea at random
  - (cf. David Wagner at Berkeley, Alex Aiken at Stanford, etc.)

Focus on Strings

- Most important buffer overrun exploits are through **string** buffers
  - Reading an untrusted string from the network, keyboard, etc.

- Focus the tool only on arrays of characters

Idea 1: Strings as an Abstract Data Type

- A problem: Pointer operations and array dereferences are very difficult to analyze statically
  - Where does *ptr point?
  - What does buf[j] refer to?

- Idea: Model effect of string library functions directly
  - Write down the effects of strcpy, strcat, etc.

Idea 2: The Abstraction

- Model buffers as pairs of integer ranges
  - *Alloc* min allocated size of the buffer in bytes
  - *Used* max number of bytes actually in use

- Recalling our Dataflow Analysis for Null or Constant values, we'll use integer ranges
  - [x,y] = { x, x+1, …, y-1, y }
  - Alloc and used cannot be computed exactly (undecideable) but can be approximated

The Strategy

- For each program expression, write **constraints** capturing the alloc and used of its string subexpressions
  - Recall constraints from Test Input Generation, where they appeared as path conditions
- Solve the constraints for the entire program

- Check for each string variable s

$$\textbf{used(s)} \leq \textbf{alloc(s)}$$

The Constraints

char s[n];	n = alloc(s)
strcpy(dst,src)	used(src) $\leq$ used(dst)
p = strdup(s)	used(s) $\leq$ used(p) & alloc(s) $\leq$ alloc(p)
p[n] = '\0'	min(used(p),n+1)) $\leq$ used(p)

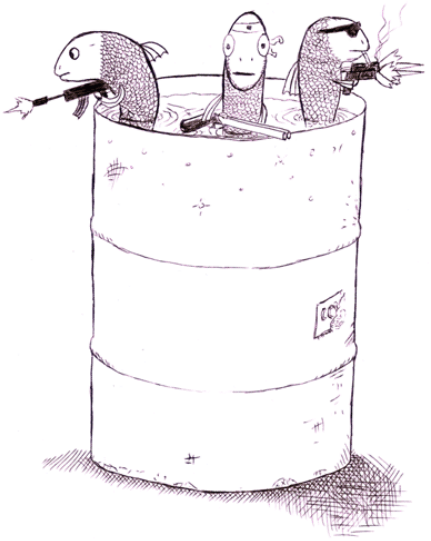
Constraint Solving

- We can solve these constraints by building on the techniques we used for dataflow analysis
  - (Recall Liveness and our Null Pointer / Constant analyses)
- Build a graph
  - Nodes are len(s), alloc(s)
  - Edges are constraints $\text{len}(s) \leq \text{len}(t)$
- Propagate information forward through the graph
  - Special handling of loops in the graph

Results

- This technique found new buffer overruns in *sendmail*
  - Which is like “shooting fish in a barrel” …
- Found new exploitable overruns in Linux *nettools* package

- Both widely used
- Previously hand-audited packages



Limitations

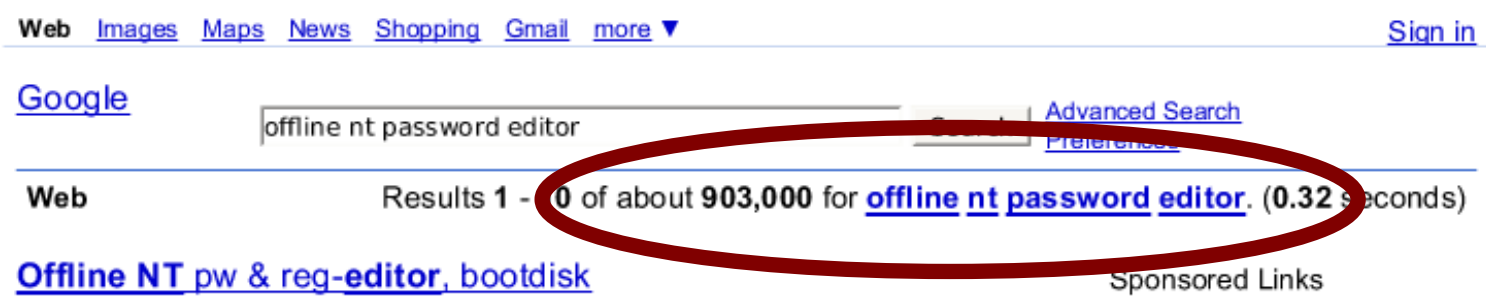
- Tool produces many **false positives** (*why?*)
  - 1 out of 10 warnings is a real bug
- Tool has false negatives (*why?*)
  - Unsound: may miss some overruns
- But still productive to use (*when?*)

- (This “put it all together” slide is the thing to think about when studying for an exam.)

Physical Security

- It is generally accepted that anyone with physical access to a machine (i.e., anyone who can open the case) can *compromise that entire machine*.
- Given physical access ...
  - How would I read your personal files?
  - How would I leave a backdoor (rootkit) for myself?
  - How would I log in as you?
- Ignore networked/encrypted filesystems for now ...

A Fairy Tale? Not Quite.



Web Images Maps News Shopping Gmail more ▼ Sign in

Google offline nt password editor Advanced Search Preferences

Web Results 1 - 10 of about 903,000 for offline nt password editor. (0.32 seconds)

Offline NT pw & reg-editor, bootdisk
Offline NT Password & Registry **Editor**, Bootdisk / CD ... Tested on: **NT** 3.51, **NT** 4 (all versions and SPs), Windows 2000 (all versions & SPs), Windows XP (all ...
home.eunet.no/pnordahl/**nt**passwd/bootdisk.html - 12k - Cached - Similar pages

Sponsored Links

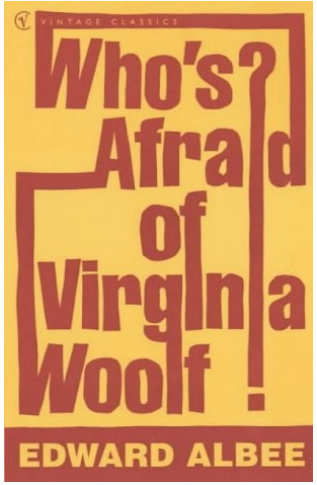
Active@ **Password** Changer
Reset passwords XP Vista 2003 2000 DOS & Win boot disk. Download now!
www.**Password**-Changer.com/

Offline NT Password & Registry **Editor**
Forgot your **NT** admin **password**? Reinstall? Oh no... But not any more. ... It works **offline**, that is, you have to shutdown your computer and boot off a ...
home.eunet.no/pnordahl/**nt**passwd/ - 1k - Cached - Similar pages
More results from home.eunet.no »

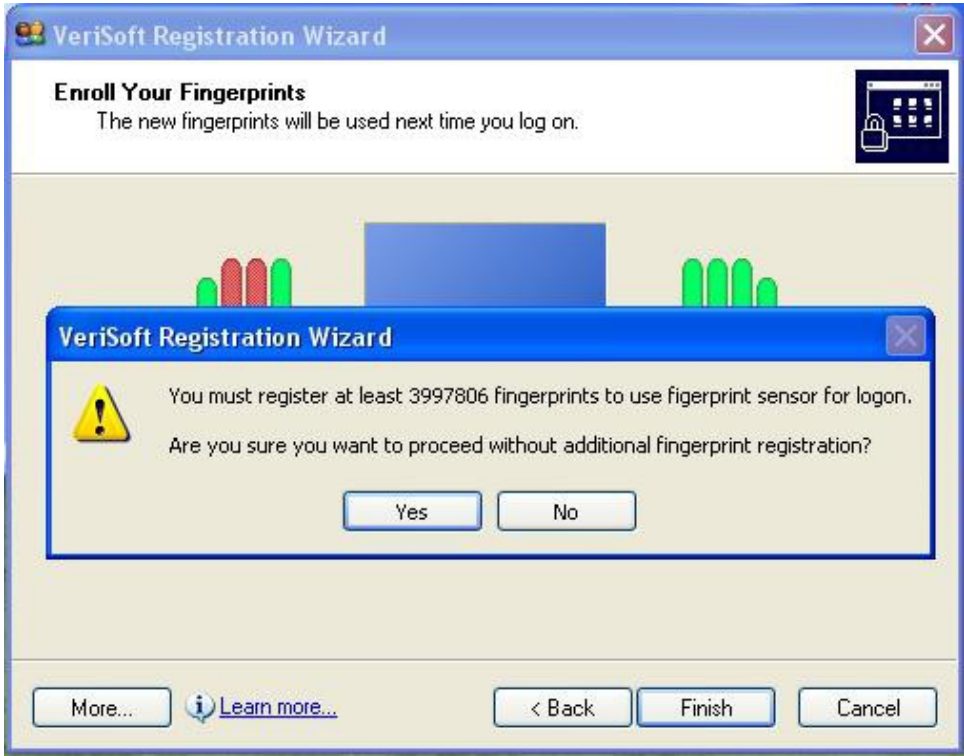
Lost or forgotten Windows **NT** / 2000 / XP **password**.
The **offline NT password** & registry **editor** is a great utility that enables users to overwrite their Windows **NT**, 2000, and XP SAM file, the file containing ...
www.computerhope.com/issues/ch000172.htm - 13k - Cached - Similar pages

Offline NT Password and Registry **Editor**
Offline NT Password and Registry **Editor** is a utility for setting or resetting the **password** of any user that has a valid (local) account on your **NT** system. ...
searchwindowssecurity.techtarget.com/downloadPage/0,295339,sid45_gci1115030,00.html - 42k - Cached - Similar pages

Hey You!
Get Off Of My Lawn!



- Must keep people out of the server room ...
- Heavy-weight physical security measures are often skipped entirely
- They are “not worth it” to the people involved
- ***Social engineering***

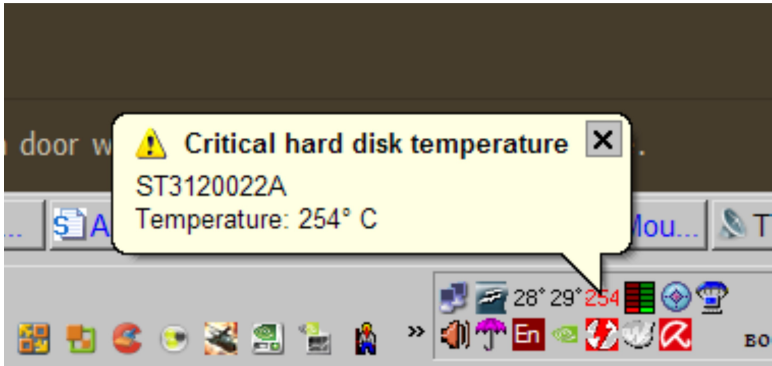


Corporate Espionage

- The cost of local corporate espionage is $1.1 trillion annually vs. $0.4 trillion for critical data stolen remotely (G4S, for 2019+)
- Office card keys (“no drafting”) and dumpster-diving prevention are two Top Five ways to defeat espionage
- ***Social engineering awareness*** is much more important, however!

Death By Heat Lamp?

- Sophisticated physical attacks are possible
  - S. Govindavajhala and A. Appel: Using Memory Errors to Attack a Virtual Machine. *IEEE Symposium on Security and Privacy*, 2003
- They write a Java program that can break out of the Java Virtual Machine if a single bit error occurs in memory ...
  - Shine lamp on memory!
- For the rest of this talk I'll assume physical security.



Is Unix Any Better?

- No; if you have physical access to a unix machine you can get root access.
  - Linux example: reboot, wait for GRUB/LILO, ask for the bootloader prompt, and type:

    `linux init=/bin/bash`
- One solution: store important files on encrypted (sub-)filesystem
  - Either requires frequent password entry or stores password in memory
  - This is only secure if no malicious programs run
  - Thus: we still need operating system security!

Unix Security Model

- All files in Unix filesystems have ***permissions***
  - **-rwxr-xr-x** 1 root root 735004 2008-01-15 09:29 /bin/bash
- Three levels: user, group, others
- Exception: a special ***root*** user can change the permissions on any file (and thus do anything)
- Passwords must be stored for login to work
- Password file stores ***hashes***:
  - smt6k:SASsHTBDJKdsa4:510:511:Sean Talts:/home/smt6k:/bin/bash
  - eas2h:p3612PxZBAx37ne:511:513:Elizabeth Soechting:/home/eas2h:/bin/bash
  - dsn9m:aw73sXHa3I3dn348:512:514:David Noble:/home/dsn9m:/bin/bash

Trojan Horses

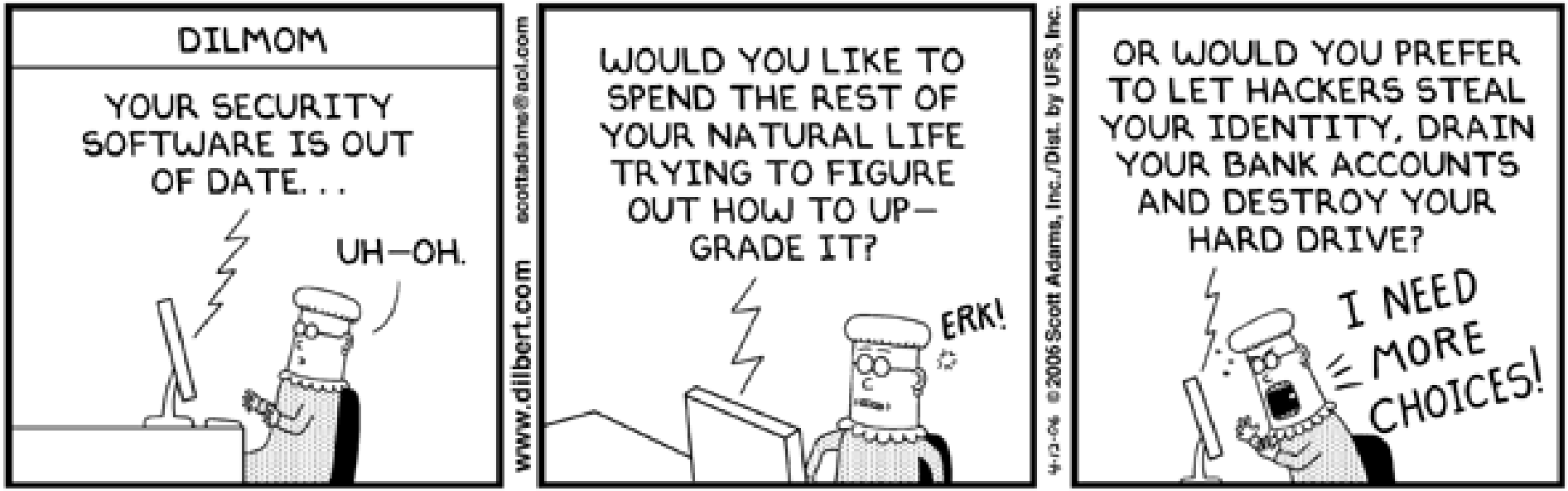
- root is convenient ... but also dangerous!
- Suppose you are running out of disk space and are hunting around for files to remove
  - Evil user makes evil files called “ls” and “dir”
  - These trojan horses email your password to Microsoft *and then* list the files
- **This single concept accounts for the vast majority of Windows vulnerabilities**
  - Pre-Vista you were always “root”, so if I could get you to click on some evil program I send over the network, I could take over your computer.

Detecting Malicious Programs

- So we need to detect viruses/trojans/worms
- This is done by regular expressions (really)
- A virus or trojan typically leaves most of the program unchanged (to avoid suspicion) and tacks on a special ***payload*** for dirty work
- Make one regular expression for each payload
  - Called the virus ***signature***
- Scan programs with union of regexps
  - A virus database file is basically just huge list of regular expressions

Escalation

- One key problem with this approach is that you must constantly update your database in response to new virus inventions
- cf. are post-release changes easy or hard?



Q: Events (597 / 842)

- Identify the speaker: *"This is a court of law, young man, not a court of justice."* and *"I have no respect for the passion of equality, which seems to me merely idealizing envy."*

Q: Games (536 / 842)

- These 1912 ring-shaped hard candies traditionally came in five flavors and were packaged in "rolls" of fifteen pieces.

Real-World Languages

- This tonal Indo-Aryan language boasts over 130 million speakers, mostly in north western India and eastern Pakistan. Its English name comes from the Persian “five waters” (*panj ab*), a reference to the Indus river. It has a canonical subject-object-verb word ordering and uses postpositions. Nouns feature two genders, two numbers, and five cases.
  - Example: ਲਹੌਰ ਪਾਕਿਸਤਾਨ ਪੰਜਾਬ ਦਾ ਦਾਰੁਲ ਹਕੂਮਤ ਐ। ਲੋਕ ਗਿਣਤੀ
  - Example: وسدا اے۔ ایسدی لوک گنتی اک کروڑ دے نیڑے اے۔

Q: Games (572 / 842)

- Which of the following mythical creatures cannot traditionally turn people to stone?
  - Basilisk
  - Cockatrice
  - Golem
  - Gorgon

Real-World Languages

- This West Germanic language features about 400 million native speakers. It is strongly stressed, uses minimal inflection, and an almost-exclusive SVO word ordering. Vocabulary choices are strongly influenced by French, Latin and Germanic roots. Writing is rendered using a Latin script; orthography is not phonemic.
- Hint: It is the most spoken language in the world (if you separate Mandarin from Cantonese, etc.).

Does This Work?

- Assume we've solved the update problem.
- What could go wrong with searching for exact code sequences?

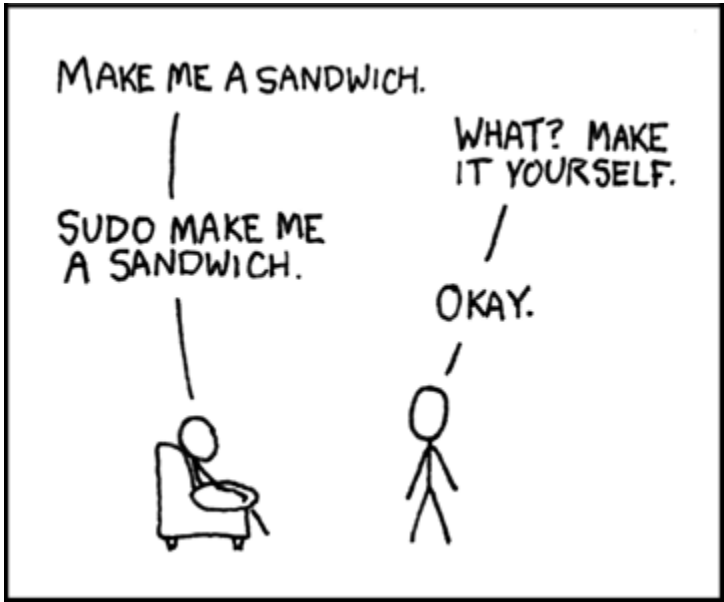
Stealth

- Any change to the virus defeats the signature
- Beware: self-modifying virus!
- ***Encryption*** with a new key per file
  - payload = decrypt module + encrypted virus code
- ***Polymorphic*** Virus: new decrypt per file
  - payload = unique decrypt + encrypted virus code
- ***Metamorphic*** Virus: rewrite each time
  - Basically: insert no-ops, “optimize” virus, etc.
  - Win32/Smile is >14000 lines of ASM, 90% of which is metamorphic engine ... and was out in 2002

My Secret Identity

- If you know another user's password, you can become that user (i.e., substitute its userid for yours --- like logging in as that person)
- The ***su*** and ***sudo*** programs implements this

Using a root account is rather like being Superman; an administrator's regular user is more like Clark Kent. Clark Kent becomes Superman for only as long as necessary, in order to save people. He then reverts to his "disguise". Root access should be used in the same fashion. The Clark Kent disguise doesn't really restrict him though, as he is still able to use his super powers. This is analogous to using the sudo program.



Design Principle / Pattern

- The **Principle of Least Privilege** (or principle of minimal privilege or principle of least authority), requires that in a particular abstraction layer of a computing environment, every module (such as a process, a user, or a program) must be able to access only the information and resources that are necessary for its legitimate purpose
- When designing software, separate the notion of a *user account* from a *role*.

A Sendmail Dilemma

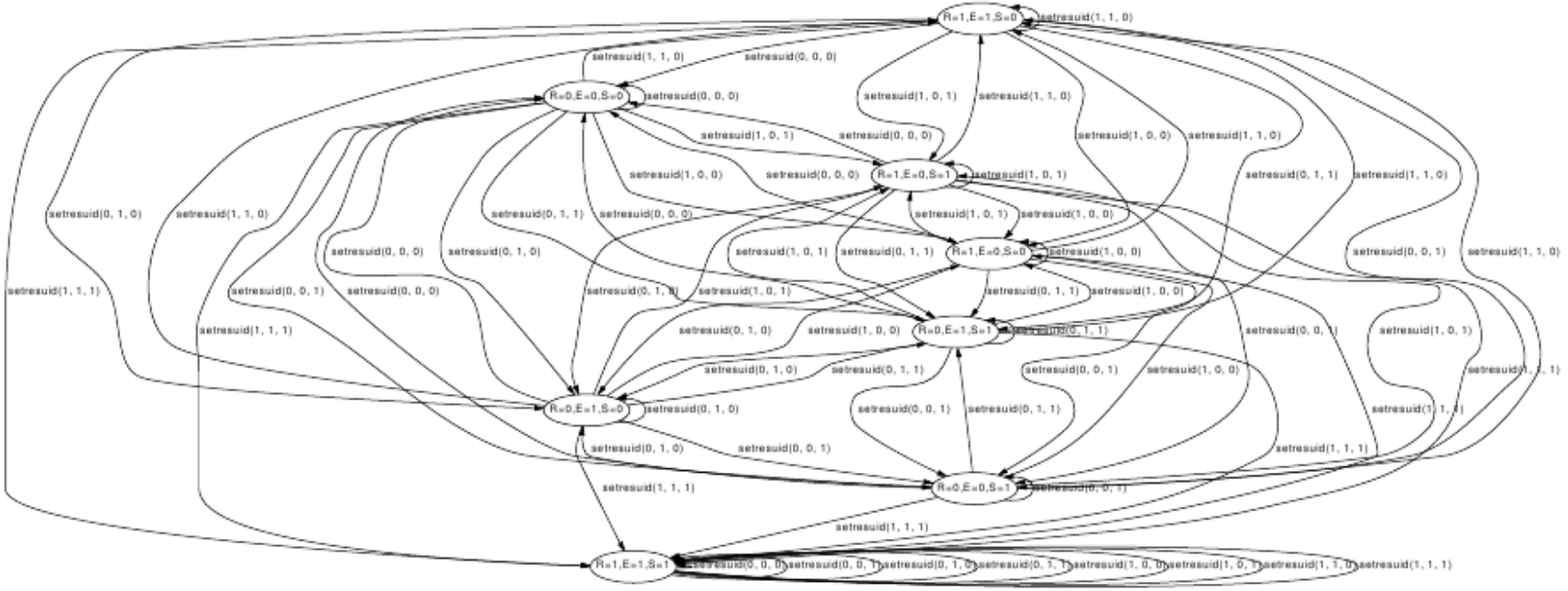
- Some programs, such as **sendmail**, must run as root to do useful work
  - Mail programs must be able to append incoming mail to the end of a given user's mailbox file
- These programs also do less-critical work
  - Mail programs may run a user-specified “vacation” program that responds to mail with “I'm away for two weeks”-style messages
- Any possible problems?

Dropping Privileges

- Important system tasks that must run as root try to drop those privileges as quickly as possible
  - Sendmail appends incoming mail to your inbox, then throws away its super powers, then runs your vacation program
- However, if you have a buffer overrun (or somesuch) I may be able to trick you into doing something before you drop privileges

Setuid Demystified

- Dropping privileges correctly is tricky, but that's another story ... [Chen, Wagner, Dean. *Usenix '02*]



(c) An FSA describing *setresuid* in Linux

Figure 5: Three finite state automata describing the *seteuid*, *setreuid*, *setresuid* system calls in Linux respectively. Ellipses represent states of the FSA, where a notation like "R=1,E=0,S=1" indicates that $euid = 0$ and $ruid = suid \neq 0$. Each transition is labelled with the system call it corresponds to.

Leaking Information

- Consider this version of login: what's wrong?

```
let name = recv_from_network () in
let pword = recv_from_network () in
let file = open_in (“/etc/passwd”) in
while not end_of_file(file) do
  let name2, hpword2 = read_from (file) in
  if name = name2 then
    return (hash(pword) = hpword2)
done ;
return false
```

Side-Channel Attacks

- Imagine it takes ***t*** microseconds to read in the entire password file
  - Then it takes ***t*** microseconds to return false for a made-up username
  - But ***t/2*** microseconds (on average) to return false for a real username with a bad password
- A ***side-channel attack*** is any attack based on information gained from the implementation of a cryptosystem, *not* from a theoretical weakness
  - Examples: timing info, power consumption, electromagnetic leaks, Spectre, Meltdown

spam bacon sausage ...

- Not everyone is running a server that I can exploit ... how can I get a payload to you?
- ***Spamming*** is abusing an electronic messaging system (i.e., email) to send unsolicited bulk messages.
- Started in the mid-1990s, spam now accounts for 80-85% of all email in the world (conservative) to as much as 95% of all world email.
  - Cost of spam estimated at $20 Billion every year, with 85% of organizations targeted by phishing scams in 2020+
- Today most spam is sent from ***zombie*** networks of virus-infected machines

Why does spam work?

- Based on physical-world direct mail, bulk mail, targeted marketing, etc.
  - Like mailed advertising with grocery coupons
  - Those work because you can get huge amounts of statistical information just from the zip code
  - ... and because people go to nearby supermarkets
- Example: if you live in 48109
  - AGI of $32,020, 69.7% white, 5.7% Black, 3.8% Hispanic, average commute 11.8 minutes, …
- Bulk physical mail is ***not*** a shot in the dark
  - Benefit (medium) exceeds cost (low)

SPAM

- Spam also works because of a cost-benefit analysis
  - Benefit (mico)
  - Cost (none) *(why?)*
- Ultimately, *some people click on spam*.
  - Not just ***phishing*** spam either!

Harvesting

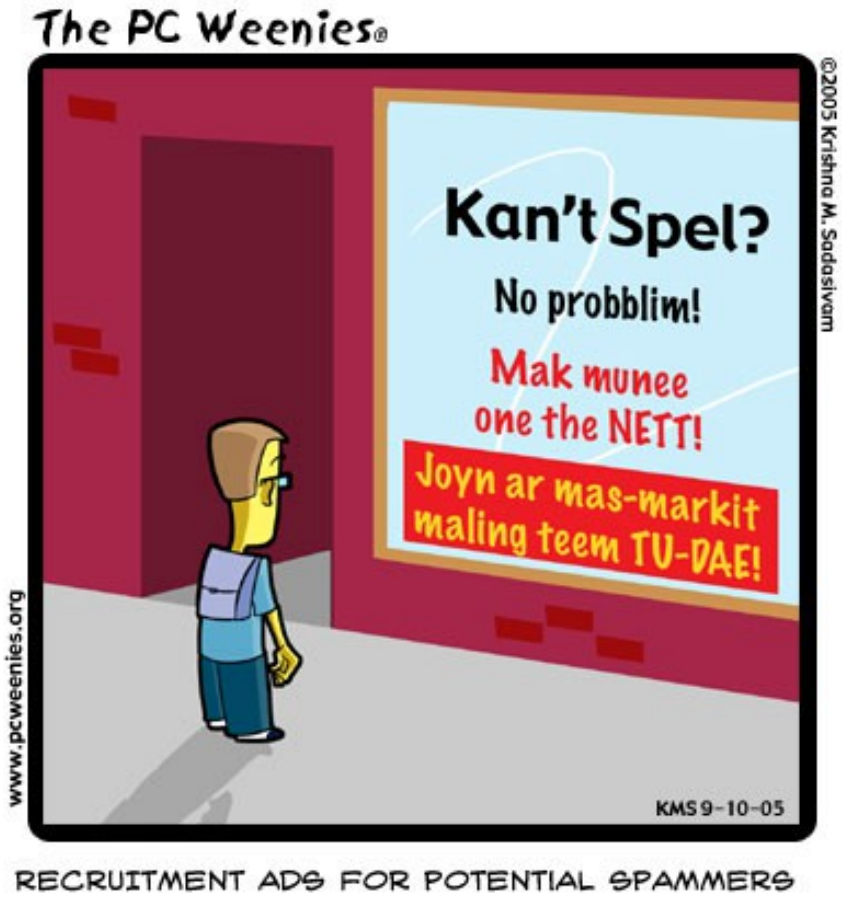
- How do I get a list of email addresses?
- ***Dictionary Spamming***
  - Guess by using a dictionary of plausible names as prefixes to known (registered) domain names
- ***Spambot Web Crawling***
  - Gather from web sites, newsgroups, special-interest group postings, chat-room conversations
  - Basically, regular expressions!
- Selling email lists is a big business ...

Stopping Spam

- ***Blocklisting*** (or "blacklisting") – do not accept messages from domain X?
  - Defeated by zombie botnets, remailers, ...
- How to find domain X?
  - Wait for users to report it ...
  - ***List poisoning***: subscribe fake "honeypot" email addresses to mailing lists, post them on web: any email that gets to them is spam
- Other, more technical approaches (e.g., greylisting), but mostly ...

Filtering

- ***Filtering*** – examine the contents of an email message and try to predict mechanically if it is spam or not
  - Simplest approach: block words (e.g., viagra)
  - Easily thwarted: (v1agra)
  - More complex: bayesian network filtering ...



SPAM Solutions

- Ultimate problem is that sending email is free
  - The Tragedy of the Commons *(read on Wikipedia)*
- SMTP, the current mail protocol, is an entrenched legacy problem
- Thus only incremental solutions are viable
- Training models to discriminate between spam and valid email is an open area of research!
- Crackpot solutions are a dime a dozen, as we can see by this idea rejection simple chart ...

Your post advocates a

( ) technical ( ) legislative ( ) market-based ( ) vigilante

approach to fighting spam. Your idea will not work because:

( ) Spammers can easily use it to harvest email addresses

( ) Mailing lists and other legitimate email uses would be affected

( ) No one will be able to find the guy or collect the money

( ) It is defenseless against brute force attacks

( ) It will stop spam for two weeks and then we'll be stuck with it

( ) Users of email will not put up with it

( ) Microsoft will not put up with it

( ) The police will not put up with it

( ) Requires too much cooperation from spammers

( ) Requires immediate total cooperation from everybody at once

( ) Many email users cannot afford to lose business or alienate potential employers

( ) Spammers don't care about invalid addresses in their lists

( ) Anyone could anonymously destroy anyone else's career or business

Specifically, your plan fails to account for:

( ) Laws expressly prohibiting it

( ) Lack of centrally controlling authority for email

( ) Open relays in foreign countries

( ) Ease of searching tiny alphanumeric address space of all email addresses

( ) Asshats

( ) Jurisdictional problems

( ) Unpopularity of weird new taxes

( ) Public reluctance to accept weird new forms of money

( ) Huge existing software investment in SMTP

( ) Willingness of users to install OS patches received by email

( ) Armies of worm riddled broadband-connected Windows boxes

( ) Eternal arms race involved in all filtering approaches

( ) Extreme profitability of spam

( ) Joe jobs and/or identity theft

( ) Technically illiterate politicians

( ) Extreme stupidity on the part of people who do business with spammers

( ) Dishonesty on the part of spammers themselves

( ) Bandwidth costs that are unaffected by client filtering

( ) Outlook

and the following philosophical objections may also apply:

( ) Ideas similar to yours are easy to come up with, yet none have ever been shown practical

( ) Any scheme based on opt-out is unacceptable

( ) SMTP headers should not be the subject of legislation

( ) Blacklists suck

( ) Whitelists suck

( ) We should be able to talk about Viagra without being censored

( ) Countermeasures should not involve wire fraud or credit card fraud

( ) Countermeasures should not involve sabotage of public networks

( ) Countermeasures must work if phased in gradually

( ) Sending email should be free

( ) Why should we have to trust you and your servers?

( ) Incompatiblity with open source or open source licenses

( ) Feel-good measures do nothing to solve the problem

( ) Temporary/one-time email addresses are cumbersome

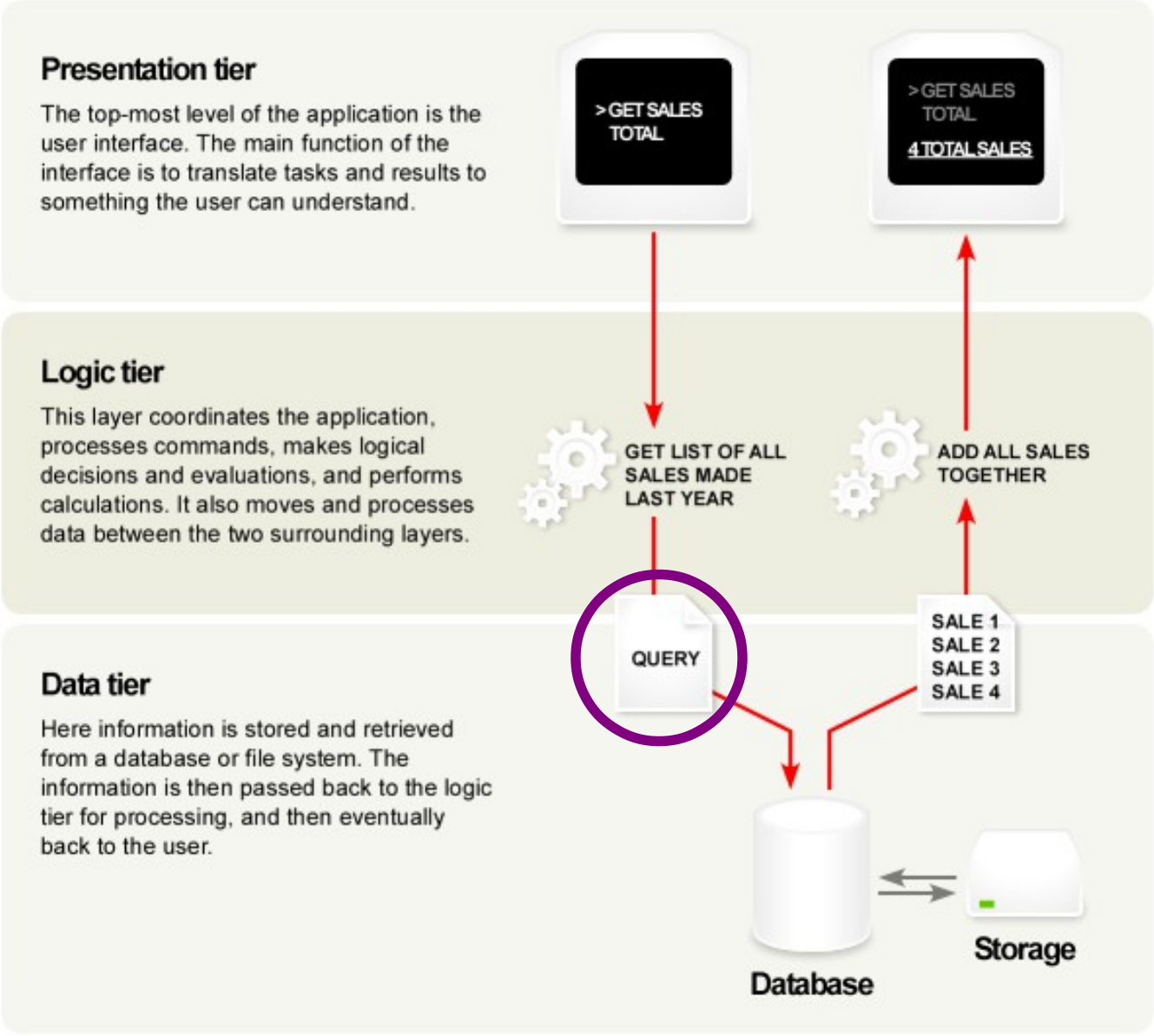
( ) I don't want the government reading my email

Cat and Mouse

- Suppose I have a server (e.g., Amazon.com)
- Let's imagine that I have solved ...
  - Viruses: no malicious code on machine
  - Buffer overruns: no injection of evil code (etc.)
  - Privileges: no running as root
  - Spam: as long as I'm dreaming, I'd like a pony ...
- I can still convince the server to do the wrong thing with the resources it legitimately has access to ...

Three-Tier Web Application

- This is how Amazon is structured
- Query is a SQL database command generated by program logic



The Problem In The Logic Tier

```
$userid = read_from_network();

if (!eregi('[0-9]+', $userid)) {
  unp_msg('You entered an invalid user ID.');
  exit;
}

$user = $DB->query("SELECT * FROM `unp_user`".
                   "WHERE userid='$userid'");

if (!DB->is_single_row($user)) {
   unp_msg('You entered an invalid user ID.');
   exit;
}
```

The Problem

```
$userid = read_from_network();

if (!eregi('[0-9]+', $userid)) {
   unp_msg('You entered an invalid user ID.');
   exit;
}

$user = $D                                    user`".
                                              id'");

if (!DB->is_single_row($user)) {
    unp_msg('You entered an invalid user ID.');
    exit;
}
```

Matches any string that contains a sequence of digits...

The Bad Place

```
// $userid == "1'; DROP TABLE unp_user; --"

if (!eregi('[0-9]+', $userid)) {
  unp_msg('You entered an invalid user ID.');
  exit;
}

$user = $DB->query("SELECT * FROM `unp_user`".
                   "WHERE userid='$userid'");

if (!DB->is_single_row($user)) {
   unp_msg('You entered an invalid user ID.');
   exit;
}
```

The Bad Place: Destroying Data

```
// $userid == "1'; DROP TABLE unp_user; --"

if
    SELECT * FROM `unp_user`
                WHERE userid='1';
    DROP TABLE unp_user;
}
    --'

$user = $DB->q                "SELECT * FROM `unp_user`".
                              "WHERE userid='$userid'");

if (!DB->is_single_row($user)) {
   unp_msg('You entered an invalid user ID.');
   exit;
}
```

Also A Bad Place: Viewing Data

```
// $userid == "1' OR 1 = 1 --"

if
                                                    ;

}

$user = $DB->q                  "SELECT * FROM `unp_user`".
                                "WHERE userid='$userid'");

if (!DB->is_single_row($user)) {
   unp_msg('You entered an invalid user ID.');
   exit;
}
```

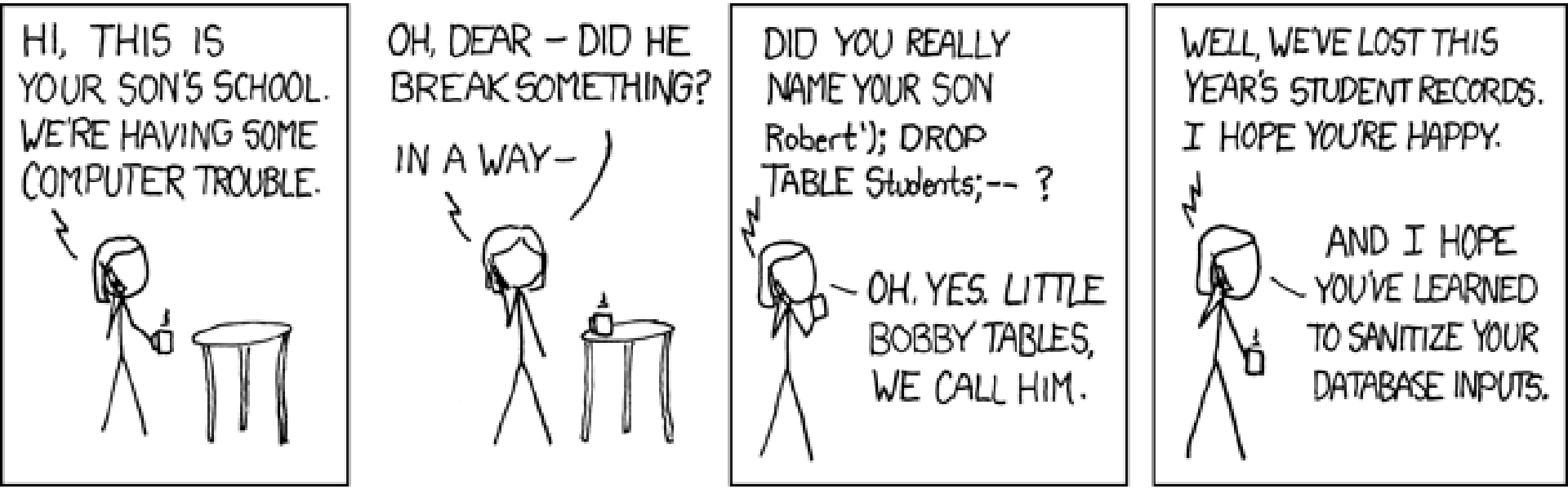
```
SELECT * FROM `unp_user`
          WHERE userid='1'
          OR 1 = 1
--'
```

SQL Code-Injection Vulnerabilities

- A ***SQL injection*** attack exploits a vulnerability in the database layer of an application whereby user input is incorrectly filtered for string literal escape characters or otherwise unexpected executed.
- Very common type of vulnerability (2006+)
- Attacks are easy and expose valuable data

Exploits Of A Mom

- The essence of SQL injection:



Cross-Site Scripting

- ***Cross-Site Scripting*** (XSS): the same flavor
- Evil X posts a message with evil JavaScript in it (e.g., send passwords to me) to Blog B
  - Blog B can also be a Piazza forum, etc.
- Later, Luser browses Blog B
- Blog B sends over data, including Evil X's Message
- Luser thinks it is from Blog B (misplaced trust)
- Luser's browser renders and interprets it

Stopping Evil Posts

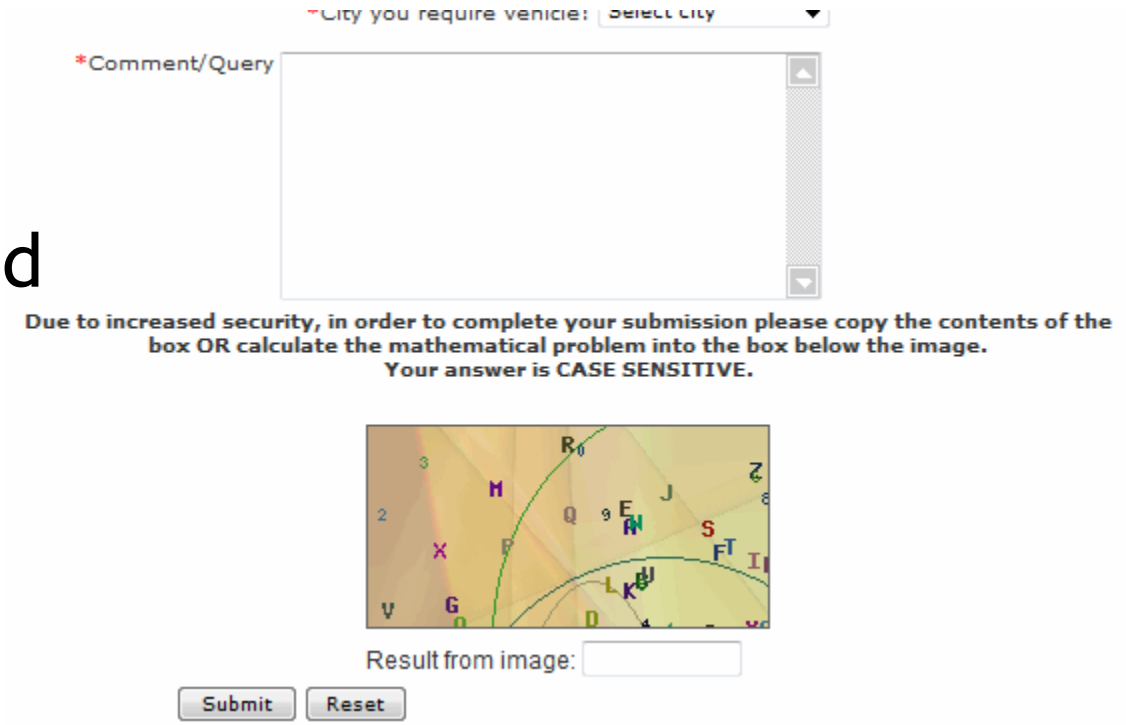
- Evil network-crawling robots try to post evil JavaScript to every forum they can find
- Let's require a real human when posting
- Increases cost
- ***CAPTCHA***

  Complete Automated

  Public Turing test

  to tell Computers

  and Humans Apart



Have We Won Yet?

- CAPTCHAs fail in theory and in practice
- The overarching problem is exactly the same:
  - The server takes input from an untrusted user
  - That input may be interpreted by another parser later
    - In SQL-CIVs, by the database's SQL parser
    - In XSS, by a user's JavaScript parser
  - So all of the same techniques apply for XSS
- Also, machines routinely win the Turing Test
  - http://en.wikipedia.org/wiki/Turing_test#Loebner_Prize

Homework

- Exam December 14th
- Everything Due December 16th