

**Sam Bowne**

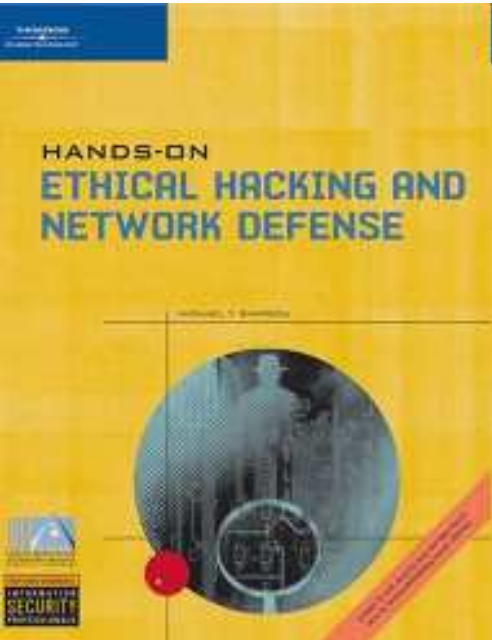
Computer Networking and Information Technology

City College San Francisco

Email: sbowne@ccsf.edu

Web: samsclass.info

## Two Hacking Classes



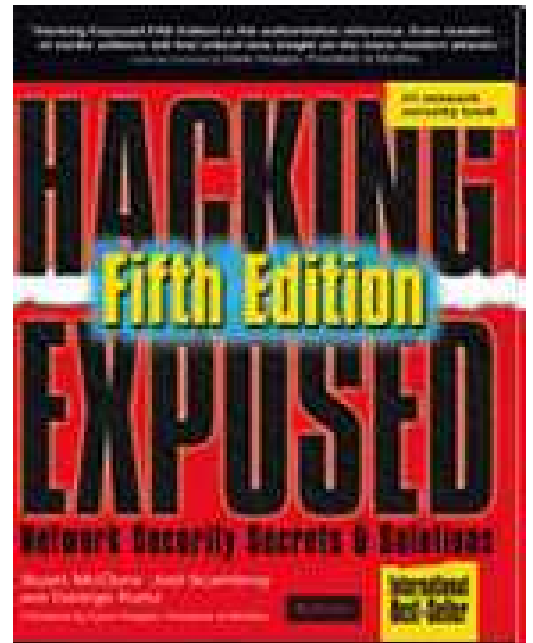
### CNIT 123: Ethical Hacking and Network Defense

Has been taught since Spring 2007 (four times)

Face-to-face and Online sections available Fall 2008

### CNIT 124: Advanced Ethical Hacking

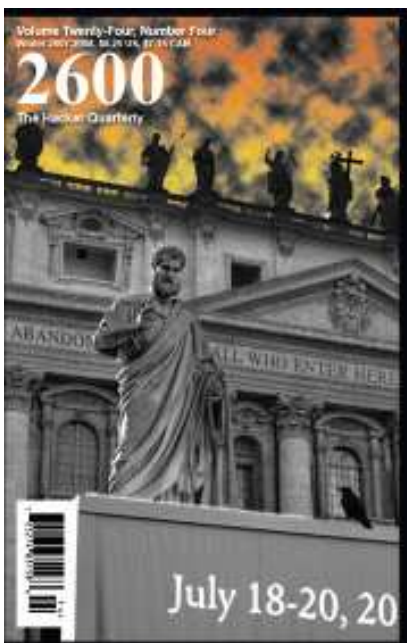
Taught for the first time in Spring 2008



## Supplemental Materials

Projects from recent research

Students get extra credit by attending conferences



## Certified Ethical Hacker

Those two classes prepare students for CEH Certification

## Certificate in Network Security Associate of Science Degree



## Four Vulnerabilities

### SQL Injection

- 16% of Web sites vulnerable

### Cross-Site Scripting

- 65% of major sites vulnerable

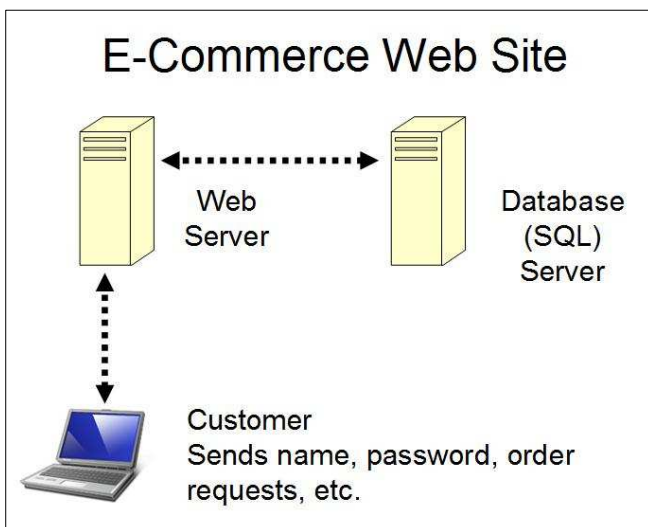
### Cross-Site Request Forgery

- Almost every Web site with a login is vulnerable

### Layer 7 Denial of Service

- Every site with active content is vulnerable

## SQL Injection



### E-Commerce Login

HTML Form collects name and password

It's processed at the SQL server with code like this:

- `SELECT * FROM customer WHERE username = 'name' AND password = 'pw'`

### SQL Injection

If a hacker enters a name of `' OR 1=1 --`



The SQL becomes:

- `SELECT * FROM customer`
- `WHERE username = '' OR 1=1 --' AND password = 'pw'`

The -- ends the statement, making the rest of the line a comment  
1=1 is always true, so this makes the condition true

## Demonstration

### SQL Injection Effects

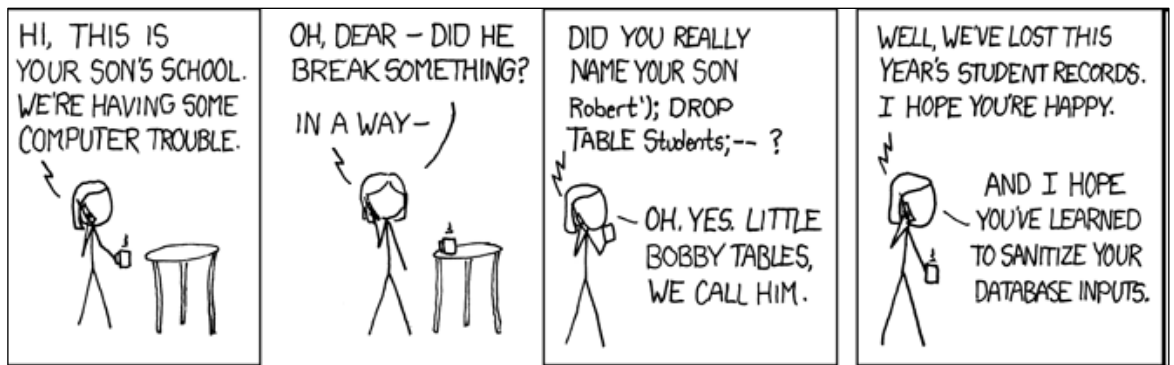
This can cause the user to be authenticated as administrator, dump the entire database, or have other drastic effects



Comic from xkcd.org

### Sanitize your Inputs

All user input should be checked, and special characters like ' or " or < or > discarded



That will reduce vulnerability to SQL injection

- The typical SQL Injection vulnerability takes more than four months to locate and fix

## Cross-Site Scripting (XSS)

### Web Message Board

#### Cross-Site Scripting (XSS)

One client posts active content, with <script> tags or other programming content

When another client reads the messages, the scripts are executed in his or her browser

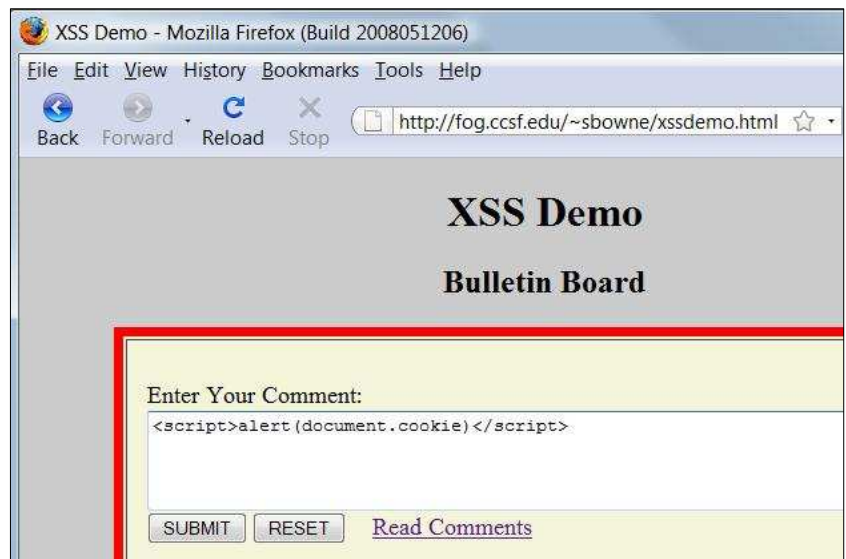
One user attacks another user, using the vulnerable Web application as a weapon

#### Demonstration

```
<script>alert("XSS vulnerability!")</script>
```

```
<script>alert(document.cookie)</script>
```

```
<script>>window.location="http://www.ccsf.edu"</script>
```



## XSS Scripting Effects

Steal another user's authentication cookie

- Hijack session

Harvest stored passwords from the target's browser

Take over machine through browser vulnerability

Redirect Webpage

Many, many other evil things...

## Cross-Site Request Forgery (XSRF)

### Web-based Email

#### Cross-Site Request Forgery (XSRF)

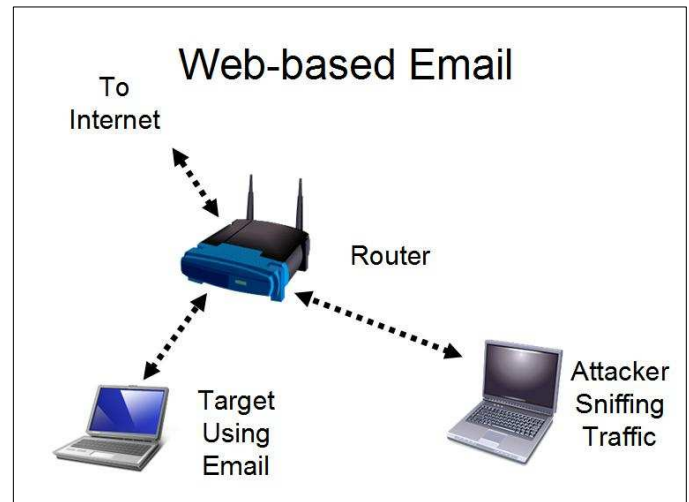
Gmail sends the password through a secure HTTPS connection

- That cannot be captured by the attacker

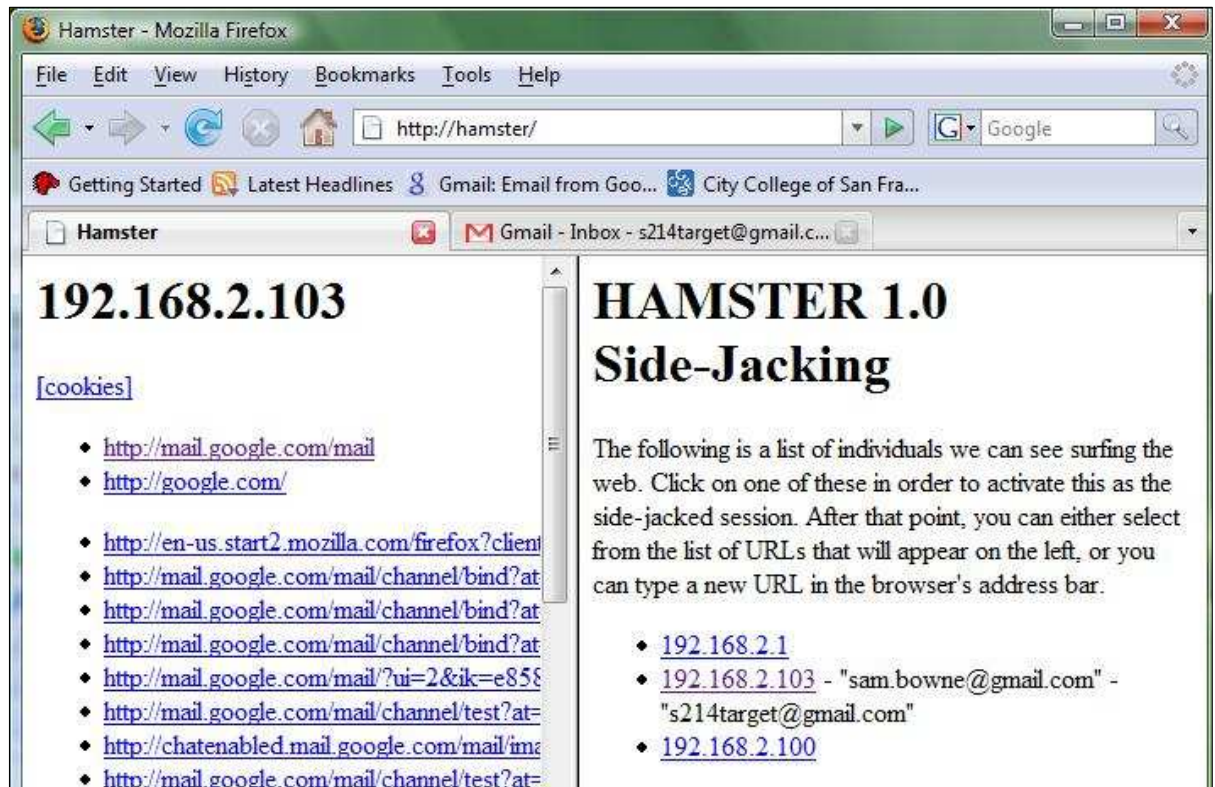
But the cookie identifying the user is sent in the clear—with HTTP

- That can easily be captured by the attacker

The attacker gets into your account without learning your password



## Demonstration



## XSRF Countermeasure

Use <https://mail.google.com> instead of <http://gmail.com>

No other mail service has this option at all, as far as I know

## Application-Layer Denial of Service

### Application-Layer DoS

Find small requests that consume a lot of server resources

Application Crashing

Data Destruction

Resource Depletion

- Memory
- CPU
- Bandwidth
- Disk Space

### Resource Depletion Example

CPU Consumption

- On a large forum
- Create a complicated regular expression search
- Use a script to launch the search over and over

### Real-World Test

Hacktics, a security company, brought down a large corporate network with just three laptops in an authorized test

- Global company with branches in Israel, Europe and the USA
- Internet Connectivity – 3x50Mbps lines with load balancing. ISPs provide Cisco (Riverhead) based Anti DDoS solutions
- High security network, 30+ Web servers, backend servers, Mail Relay, databases

### Hacktics Results

DoS was successful to all systems but one

Two applications crashed completely after a few dozen requests only

Most other applications stopped responding after 5-15 minutes of script execution from up to three laptops (though with most a single laptop was sufficient)

Main cause of DoS was CPU exhaustion

### References

Where the Web is Weak

- [http://www.forbes.com/2008/05/14/web-hacking-google-tech-security08-cx\\_ag\\_0514webhack.html](http://www.forbes.com/2008/05/14/web-hacking-google-tech-security08-cx_ag_0514webhack.html)

Application-Layer DDoS Attacks

- <http://networks.rice.edu/papers/2006-04-Infocom-final.ppt>