Web Security

Ben Adida
CIS, CSAIL, MIT

6.171
9 May 2006
“Wisdom consists in being able to distinguish among dangers and make a choice of the least harmful”

Machiavelli
Threat Model

- Who is the attacker?
- Who is being attacked?
- What constitutes a breach?
- What is the cost of a breach?
- What is the response plan?
Remember

• Every Page is a Program
  *Inputs provided by attacker*
  *Attacker is not constrained by browser*

• You’re Not Just Protecting Yourself
  *You’re protecting your user*

• Browsers Assume you know what you’re doing
  *HTML + code delivered by you*
“Using encryption on the Internet is the equivalent of arranging an armored car to deliver credit card information from someone living in a cardboard box to someone living on a park bench.”

Gene Spafford
OS
<table>
<thead>
<tr>
<th>Database</th>
<th>OS</th>
</tr>
</thead>
</table>

Password Cracks

- OS
- Database
- Web Server
Password Cracks → Application Code → Web Server → Database → OS → Buffer Overflow
Harvard rejects 119 accused of hacking

Applicants' behavior 'unethical at best'

By Robert Weisman, Globe Staff | March 8, 2005
Not Checking Permissions

Harvard rejects 119 accused of hacking
Applicants' behavior 'unethical at best'
By Robert Weisman, Globe Staff | March 8, 2005

Business ethics 101: in the event of an blatant IT failure, blame the user.
Faulty Inputs
Faulty Inputs

http://acme.com/view-user?id=4
Faulty Inputs

http://acme.com/view-user?id=4

select name, email from users where USERID = $id;
Faulty Inputs

http://acme.com/view-user?id=4

select name, email from users
where USERID = $id;

view-user?USERID=NULL union select password as name, email from users
where USERID=4
Faulty Inputs

http://acme.com/view-user?id=4

```
select name, email from users
where USERID = $id;
```

```
view-user?USERID=NULL union select
password as name, email from users
where USERID=4
```

```
select name, email from users
where USERID = NULL union select password
as name, email from users where USERID=4;
```
Spoofed Inputs
Spoofed Inputs

<form action="buy-tickets">
<input type="hidden" name="adult_price" value="9.5" />
...
</form>
<form action="buy-tickets">
<input
  type="hidden"
  name="adult_price"
  value="9.5" />
...
</form>

Cookie: admin=1;
XSS - The Biggie

Web Server

Alice

Bob

<script>
document.cookie
</script>
Prevention

- Standardize Input Processing
e.g. bind variables, HTML filtering

explore your own tools...

- Build in Role-Based Action Control Checks
even if you don’t implement them yet.
e.g. user_can_edit_msg($user_id,$msg_id)
Mitigation

• Users’ Passwords
  • store them hashed and salted
  • protect your users!

• Credit Card Numbers
  • do you really need to store them?
  • consider moving them off the main server
Let’s Crack
• Nice Job on the login engine 
  you reused code.

• Nice Job on the role-based permissions 
  this is going to be very helpful to you.

• Ouch on the Cross-Site Scripting 
  you’re not even checking the <SCRIPT> tag. 
  same for Texas4000.
A Word on .NET

• It checks for XSS attacks
  you are all safe....
  until you start adding features