Passwords in MySQL (and other misc.)

CS518

Dr. Justin F. Brunelle
Problem

- Passwords should not be stored in clear-text
- Access to the DB gives you access to all user credentials
- Single point of failure for site security
Solution: Hash passwords

- Use a randomized per-use secret string and hash it with the password
- “Secret String” is a salt
- e.g.:
  - sha256($password . $salt)
- sha256 recommended
- sha1 comes standard with php
• Each user will have a salt
  – Salt created when user created

<table>
<thead>
<tr>
<th>username</th>
<th>Plain-text-pass</th>
<th>salt</th>
<th>Encrypted-pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>justin</td>
<td>mustangs</td>
<td>abcdefg</td>
<td>2704231b6ca155b6b477d3e52054c8ca98aeedd3</td>
</tr>
<tr>
<td>kent</td>
<td>basketball</td>
<td>123456</td>
<td>afd5616e055ed5e4e82882143a0fde8f0b574d09</td>
</tr>
<tr>
<td>taylor</td>
<td>football</td>
<td>abc123</td>
<td>1efa2622b5df3ee8a09e779f344aa4e7d2f9f72c</td>
</tr>
</tbody>
</table>
Making the passwords

$> php -a
Interactive mode enabled

php > print sha1("mustangs");
5c33647c01eaf3aa534c0e83376193a32cc86a95

php > print sha1("mustangs" . "abcdefg");
2704231b6ca155b6b477d3e52054c8ca98aeedd3
What this gets you...

- Multiple variables in authentication:
  - No plain-text storage (can't steal passwords)
  - Salt order (prepend or append)
  - Hash algorithm (md5, sha1, sha256...)

- But does not:
  - Pass passwords security from HTML!
  - Monitor POST data, form data
  - (Use HTTPS and SSL for this)
Disclaimer

- Using your home-grown authentication/encryption is a bad idea
- See password_hash(), oauth, or Kerberos
favicon

• Shows up in your browser's tab/window/title
• Small image (16x16 or 32x32)
• Most browser look for “favicon”
• Safer to force HTML reference:

```html
<!DOCTYPE html>
<html lang="en-US">
<head profile="http://www.w3.org/2005/10/profile">
<link rel="icon" type="image/png" href="http://example.com/myicon.png">
[…]
</head>
[…]
</html>
```