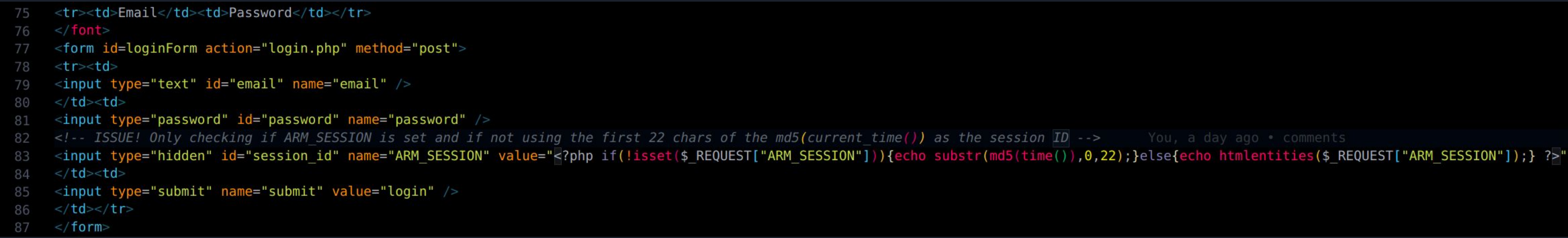
Fixing Session ID Entropy

Simon Buchheit

Problem

- This screencap shows the generation of ARM_SESSION at login



```
<tr><td>Email</td><td>Password</td></tr>
</font>
<form id=loginForm action="login.php" method="post">
<tr><td>
<input type="text" id="email" name="email" />
</td><td>
<input type="password" id="password" name="password" />
<!-- ISSUE! Only checking if ARM_SESSION is set and if not using the first 22 chars of the md5(current_time()) as the session ID -->      You, a day ago • comments
<input type="hidden" id="session_id" name="ARM_SESSION" value="<?php if(!isset($_REQUEST["ARM_SESSION"])){echo substr(md5(time()),0,22);}else{echo htmlentities($_REQUEST["ARM_SESSION"]);} ?>"
</td><td>
<input type="submit" name="submit" value="login" />
</td></tr>
</form>
```

- The issue with session ID integrity comes from generating the ID from the MD5 hash of the current time
  - MD5 is cryptographically insecure and collisions can easily be found
  - The current method allows for brute forcing of session id’s as well

Fix!

random_bytes

- Instead of generating the session using the MD5 hash, use the PHP random_bytes() function!
  - random_bytes()
    - Cryptographically secure
    - Take length as the parameter
- Link to documentation
  - https://www.php.net/manual/en/function.random-bytes.php

Additional Reminders

- Make sure the session is generated AFTER authenticating the user
  - Not when submitting the login form
  - Not from a user defined parameter