Hashing Algorithms:
The Good, the Bad, and the Fail

Steve Thomas “Sc00bz”

Passwords^12

TobTu.com/passwords12

Ranked Password Authentication

- scrypt
- bcrypt ($2y$)
- PHPass & SHA2 based crypt
- PBKDF2
- MD5 crypt (1)
- Single hash
- Almost anything with DES
- MySQL323
- XSHA1

- Selectable memory
- 4 KiB internal state
- Sequential hashing 32/64 bit
- *Never* implemented correctly
- Fixed 1002 rounds of MD5
- Fast usually unsalted
- LM, crypt DES (only 64 bits)
- Meet-in-the-middle attack
- Any 20 character PW in .5 sec

Public Service Announcement

- Why you should not use LastPass
  - Recommends 500 round PBKDF2
  - Unencrypted URL field
  - Hushmail-ish attack
- KeePass with DropBox or Google Drive
- Host your own “LastPass”

How much salt and pepper do we need?

- “4”
  - Public
    - For logging in
      - “hash(username + domain + password)”
  - “Private” salt (stored in the database)
    - No targeted precomputed attacks
  - Encryption key (stored on the auth server)
    - Database dumps are worthless
  - Site specific ROM

Site Specific ROM

- TBs of random data
- Salted password gives an offset into the ROM
- An attacker needs to download a large part of the ROM to start cracking passwords
- Easier detection of intruder
- 10 Mbps
  - max 4000 auths/s
  - >9 days/TB

What to do About This

- NIST sponsored competition for “PBKDF3”
  - Standard for storing credentials
  - Forced minimum iteration count
  - Upgrade paths without the password
    - For authentication
  - Parallel
  - Memory Hard

Parallel vs. Memory Hard

- The more parallel you are the less memory hard you are

My Best Effort

```
 1 function create_hash($pw, $salt, $count, $count2)
 2 {
 3     if ($count <= 0 || $count2 <= 0)
 4         return null;
 5
 6     $ret = hash($salt . $pw, true);
 7     // Upgrade path without the password
 8     for ($i = 0; $i < $count2; $i++)
 9     {
10         // Forced minimum iteration counts
11         $cur = hash(pack('NN', 0, $i) . $ret, true);
12         For ($j = 1; $j < 2048 * $count; $j++)
13         {
14             // Highly parallel operations
15             $cur ^= hash(pack('NN', $j, $i) . $ret, true);
16         }
17
18         $ret = hash($salt . $ret . $cur, true);
19     }
20
21     // "Standard" for storing credentials
22     return '$????$' . $count . '$' . $count2 . '$' . $salt . '$' . bin2hex($ret);
23 }
```

Pros/Cons

- Pros
  - Forced minimum iteration counts
  - Parallel
    - Authenticating with GPUs
  - Upgrade path without the password
- Cons
  - Site specific ROM
  - Not memory hard

Thank You

- Questions?