Argon2 for password hashing and cryptocurrencies

Alex Biryukov, Daniel Dinu, Dmitry Khovratovich

University of Luxembourg

20th July 2016
Recall why we need Argon2
Keyless password authentication:
  - User registers with name $l$ and password $p$;
  - Server selects hash function $H$, generates salt $s$, and stores $(l, H(s, p))$;
  - User sends $(l, p')$ during the login;
  - Server matches $(l, H(s, p'))$ with its password file.

Problems:
  - Password files are often leaked unencrypted;
  - Passwords have low entropy ("123456");
  - Regular cryptographic hash functions are cracked on GPU/FPGA/ASIC.
Dictionary attacks are most efficient on custom hardware: multiple computing cores on large ASICs.

Practical example of SHA-2 hashing (Bitcoin):
- $2^{32}$ hashes/joule on ASIC;
- $2^{17}$ hashes/joule on laptop.

ASIC-equipped crackers are the threat from the near future.

ASICs have high entry costs, but FPGA and GPU are employed too.
### Performance

<table>
<thead>
<tr>
<th>Proc.</th>
<th>Thr.</th>
<th>Argon2d (1 pass)</th>
<th>Argon2i (3 passes)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>cpb (GB/s)</td>
<td>cpb (GB/s)</td>
</tr>
<tr>
<td>i7-4500U</td>
<td>1</td>
<td>1.3</td>
<td>4.7</td>
</tr>
<tr>
<td>i7-4500U</td>
<td>2</td>
<td>0.9</td>
<td>2.8</td>
</tr>
<tr>
<td>i7-4500U</td>
<td>4</td>
<td>0.6</td>
<td>2</td>
</tr>
<tr>
<td>i7-4500U</td>
<td>8</td>
<td>0.6</td>
<td>1.9</td>
</tr>
</tbody>
</table>

**Table:** Speed and memory bandwidth of Argon2(d/i) measured on 1 GB memory filled. Core i7-4500U — Intel Haswell 1.8 GHz, 4 cores
Since 2003, *memory-intensive* computations have been proposed.

Computing with a lot of memory would require a very large and expensive chip.

With large memory on-chip, the ASIC advantage vanishes.
Argon2, the winner of Password Hashing Competition
Two variants: Argon2d and Argon2i.

- Argon2d uses data-dependent addressing ($\phi(j) = X[j - 1]$);
- Argon2i uses data-independent addressing ($\phi(j) = \text{Blake2b}(j)$);
- The block size is 8192 bits;
- The compression function is based on the Blake2b permutation, enriched with 32-bit multiplications;
- Arbitrarily level of parallelism.
Several enhancements from the version that won the PHC:

- Total memory up to 4 TB;
- Different way to take pseudo-random data for the reference block index from the previous block (Argon2i);
- In second and later passes over the memory, new blocks are XORed into old ones, not overwrite (rules out some attacks, see the last slide).
• Should there be any $H$ other than Blake2b (internally Blake2b has to stay anyway)?
• Should we allow salts shorter than 8 bytes?
• Should we restrict password hashing to Argon2i only?
Some people ask what if full SHA-3 or its internal (reduced-round) permutations is used instead of Blake2b-based one:

- Keccak permutation, 3 of 24 rounds: the same time;
- Keccak permutation, 6 of 24 rounds: 50% slower;
- Keccak permutation, 12 of 24 rounds: 2.5x slower;
- Full Keccak permutation, 24 of 24 rounds: 5x slower;
- Full SHA-3: about 10x slower.
• Collision and preimage resistance – follows from the use of full Blake2b and collision resistance of $P(x) + x$ for the internal permutation $P$.
• Tradeoff resistance assumed from public scrutiny.
Time-space tradeoff: how time grows if space is reduced.

\[ T = f\left(\frac{1}{S}\right). \]

Linear \( f \) means equal trading of space for time.

Tradeoff has attack quality \( \gamma \) if

\[ \gamma = \frac{ST}{S_{\text{new}} T_{\text{new}}}. \]

ASIC implementing this tradeoff will have advantage \( \gamma \) in time-area product (proportional to the running costs of dictionary attacks).
Timeline:

- 2014: Ranking tradeoff method (making a computing graph low-depth by storing certain vertices).
- Jan 2015: Application of ranking method to Argon2i and Argon2d.
- Jul 2015: Argon2 selected as the winner.
- Jan 2016: Corrigan-Gibbs et al. publish "optimization attack" (patched in version 1.3).
- Feb 2016: Alwen and Blocki publish a depth-reducing attack.
Attack quality – the reduction in the time-area product for Argon2-implementing ASICs. Here are ranking (2015) and other (2016) attacks on Argon2i.

<table>
<thead>
<tr>
<th>Passes</th>
<th>Ranking</th>
<th>AB 1 GB</th>
<th>AB 16 GB</th>
<th>Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>2.4</td>
<td>4.5</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>1.3</td>
<td>2.5</td>
<td>4</td>
</tr>
</tbody>
</table>

Recommended
<table>
<thead>
<tr>
<th>Passes</th>
<th>Ranking</th>
<th>AB 1 GB</th>
<th>AB 16 GB</th>
<th>Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2.5</td>
<td>0.9</td>
<td>1.8</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>0.75</td>
<td>1.4</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>0.6</td>
<td>1.2</td>
<td>-</td>
</tr>
</tbody>
</table>

Details in Section 3.6 of the Argon2 specification.
Summary of tradeoff security

Argon2d (1 pass, data-dependent):
- No generic attacks;
- Tradeoff attack: area-time product may be reduced by the factor of 1.5 (ranking method).

Argon2i (1 or 2 passes, never recommended):
- Optimization attack [Corrigan-Gibbs et al. 2016], 1/5 of memory with no penalty.

Argon2i (3 or more passes):
- Sandwich attack [Alwen-Blocki’16]: 1.8 factor for 3 passes, less than 1.4 for others.
- Ranking tradeoff attack: 2.5 factor for 3 passes.

Paranoid users can have 5-6 passes or more.