Cryptanalytic world records, summer 2014
Factoring integers

Dr. Daniel Loebenberger
<table>
<thead>
<tr>
<th>method</th>
<th>year</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>trial division</td>
<td>$-\infty$</td>
<td>$\mathcal{O} \sim (2^{n/2})$</td>
</tr>
<tr>
<td>Pollard’s $p - 1$ method</td>
<td>1974</td>
<td>$\mathcal{O} \sim (2^{n/4})$</td>
</tr>
<tr>
<td>Pollard’s $\rho$ method</td>
<td>1975</td>
<td>$\mathcal{O} \sim (2^{n/4})$</td>
</tr>
<tr>
<td>Pollard’s and Strassen’s method</td>
<td>1976</td>
<td>$\mathcal{O} \sim (2^{n/4})$</td>
</tr>
<tr>
<td>Morrison’s and Brillhart’s continued fractions</td>
<td>1975</td>
<td>$\exp(\mathcal{O} \sim (n^{1/2}))$</td>
</tr>
<tr>
<td>Dixon’s random squares, quadratic sieve</td>
<td>1981</td>
<td>$\exp(\mathcal{O} \sim (n^{1/2}))$</td>
</tr>
<tr>
<td>Lenstra’s elliptic curves</td>
<td>1987</td>
<td>$\exp(\mathcal{O} \sim (n^{1/2}))$</td>
</tr>
<tr>
<td>number field sieve</td>
<td>1990</td>
<td>$\exp(\mathcal{O} \sim (n^{1/3}))$</td>
</tr>
<tr>
<td>Shor’s quantum algorithm</td>
<td>1994</td>
<td>$\mathcal{O} (n^3)$ q.ops.</td>
</tr>
</tbody>
</table>
When $p$ is a prime, then $\mathbb{Z}_p$ is a field; in particular, it has no zero divisors. A polynomial $f$ of degree $d$ has at most $d$ roots in $\mathbb{Z}_p$. This is not true in $\mathbb{Z}_N$ when $N$ is composite. So we are looking for two values $x$, $y$ with $x^2 = y^2$ in $\mathbb{Z}_N$ but not $x \in \pm y$. This is not easy.
Let $N = 2183$. Suppose that we have found the equations

\begin{align*}
453^2 &= 7, \\
1014^2 &= 3, \\
209^2 &= 21.
\end{align*}

Then we obtain $(453 \cdot 1014 \cdot 209)^2 = 21^2$ in $\mathbb{Z}_N$, or $687^2 = 21^2$ in $\mathbb{Z}_N$. This yields the factors $37 = \gcd(687 - 21, N)$ and $59 = \gcd(687 + 21, N)$; in fact, $N = 37 \cdot 59$ is the prime factorization of $N$. 
**Algorithm.** Dixon’s random squares method.

Input: An integer $N \geq 3$, and $B \in \mathbb{N}_{\geq 2}$.
Output: Either a proper divisor of $N$, or “failure”.

1. Compute all primes $p_1, p_2, \ldots, p_h$ up to $B$.
2. If $p_i$ divides $N$ for some $i \in \{1, \ldots, h\}$ then Return $p_i$.
3. $A \leftarrow \emptyset$.
4. Repeat 5 - 11 Until $\#A = h + 1$.
5. Choose a uniform random number $b \leftarrow \mathbb{Z}_N \setminus \{0\}$.
6. $g \leftarrow \gcd(b, N)$, If $g > 1$ then Return $g$.
7. $a \leftarrow b^2 \in \mathbb{Z}_N$.
8. For $i = 1, \ldots, h$ do 9 - 10
9. 
10. While $p_i$ divides $a$ do $a \leftarrow a/p_i$, $\alpha_i \leftarrow \alpha_i + 1$.
11. If $a = 1$, then $\alpha \leftarrow (\alpha_1, \ldots, \alpha_h)$, $A \leftarrow A \cup \{(b, \alpha)\}$.
12. Find distinct pairs $(b_1, \alpha^{(1)}), \ldots, (b_\ell, \alpha^{(\ell)}) \in A$ with $\alpha^{(1)} + \cdots + \alpha^{(\ell)} = 0$ in $\mathbb{Z}_2^h$, for some $\ell \geq 1$.
13. $(\delta_1, \ldots, \delta_h) \leftarrow \frac{1}{2}(\alpha^{(1)} + \cdots + \alpha^{(\ell)})$.
14. $x \leftarrow \prod_{1 \leq i \leq \ell} b_i$, $y \leftarrow \prod_{1 \leq j \leq h} p_j^{\delta_j}$, $g \leftarrow \gcd(x + y, N)$.
15. If $1 < g < N$ then Return $g$ Else Return “failure”.
Example

We have $B = 7$, factor base $(2, 3, 5, 7)$,

\[
b_1 = 453, \ b_2 = 1014, \ b_3 = 209, \]
\[
\alpha^{(1)} = (0, 0, 0, 1), \ \alpha^{(2)} = (0, 1, 0, 0), \ \alpha^{(3)} = (0, 1, 0, 1), \]
\[
\alpha^{(1)} + \alpha^{(2)} + \alpha^{(3)} = (0, 2, 0, 2) = (0, 0, 0, 0) \text{ in } \mathbb{Z}_2^4, \]
\[
\delta_1 = \delta_3 = 0, \ \delta_2 = \delta_4 = 1, \]
\[
x = 687, \ y = 21, \text{ and } \gcd(687 - 21, N) = 37. \]

In fact, there are exactly 73 7-numbers in $\mathbb{Z}_N$, excluding 0. Thus we expect $2180/73 \approx 31$ random choices of $b$ in order to find one 7-number. We have $u = \ln 2182/\ln 7 \approx 3.95108$, $u^{-u} \approx 0.00439$, and $Nu^{-u} \approx 9.58$. This is a serious underestimate, which occurs for small values. However, 7-smoothness is the same as 10.9-smoothness, and with this value, we find $Nu^{-u} \approx 50.709$. 
Theorem

Dixon’s random squares method factors an $n$-bit integer $N$ with an expected number of

$$L_{1/2}(n)$$

operations, where $L_\alpha(n) = \exp(\mathcal{O}(n^\alpha(\log n)^{1-\alpha}))$. 
For an $n$-bit integer $N$, quantum computers can calculate orders in $\mathbb{Z}_N^\times$ using $O(n^3)$ operations on $4n$ qubits. We will now show how one can then factor $N$ efficiently.
$B_4$: Given $N = p \cdot q$, find $p$.

$B_5$: Given $N$ and $x \in \mathbb{Z}_N^\times$, compute the order $\text{ord}(x)$.

$B_5'$: Given $\epsilon \geq 0$, $N$, and $x \in \mathbb{Z}_N^\times$, either compute an integer multiple $\ell$ of $k = \text{ord}(x)$ with bit-size polynomial in that of $N$, or return “failure”; If $k$ is odd, the latter with probability at most $\epsilon$.

We clearly have $B_5' \leq_p B_5$ and we will reduce $B_4$ to $B_5'$. 
Algorithm. Reduction $A$ from $B_4$ to $B_5'$.

Input: An $n$-bit odd integer $N$, not a proper power of an integer.
Output: A proper factor of $N$, or “failure”.

1. Choose $x \leftarrow \{1, \ldots, N - 1\}$. Compute $g \leftarrow \gcd(x, N)$.
2. If $g \neq 1$ then return $g$.
3. $y \leftarrow x^{2^n}$.
4. Call an oracle for $B'_5$ to either receive a multiple $\ell$ of the order of $y$ in $\mathbb{Z}_N^\times$ or “failure”. In the latter case, return “failure”.
5. Write $\ell = 2^e m$, with nonnegative integers $e$ and $m$, where $m$ is odd.
6. $z \leftarrow x^m$ in $\mathbb{Z}_N$.
7. If $z = 1$ then return “failure”.
8. For $i$ from 1 to $n$ do 9 through 12.
9. If $z = -1$ then return “failure”.
10. $u \leftarrow z^2$ in $\mathbb{Z}_N$.
11. If $u = 1$ then compute $r \leftarrow \gcd(z - 1, N)$ and return $r$.
12. $z \leftarrow u$.
13. Return “failure”.
Example

For input $N = 21$, the 20 choices of $x$ in step 1 of Algorithm lead to the following values, where $z$ is the value in step 10.

<table>
<thead>
<tr>
<th>gcd($x, N$) ≠ 1</th>
<th>even order</th>
<th>odd order</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$  $y$  $\ell$  $k$  $z$  $r$</td>
<td>$x$  $z$  $k$</td>
<td></td>
</tr>
<tr>
<td>3, 6, 7, 9</td>
<td>2 4 3 6 8 7</td>
<td>1 1 1</td>
</tr>
<tr>
<td>12, 14, 15, 18</td>
<td>5 4 3 6 20 $f$</td>
<td>4 1 3</td>
</tr>
<tr>
<td></td>
<td>8 1 1 2 8 7</td>
<td>16 1 3</td>
</tr>
<tr>
<td></td>
<td>10 16 3 6 13 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11 16 3 6 8 7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>13 1 1 2 13 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>17 16 3 6 20 $f$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>19 4 3 6 13 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>20 1 1 2 20 $f$</td>
<td></td>
</tr>
</tbody>
</table>

The values $x$ and $y$ are from steps 1 and 3, respectively, of Algorithm, $\ell$ is the output of the order oracle, assumed to be ord($y$), so that $\ell = m$ in step 4, $k = \text{ord}(x)$, $z$ is from step 5, and $r$ is either the factor of 21 from step 10 or $f = \text{“failure”}$. Thus we obtain a proper factor of 21 for $8 + 6 = 14$ values of $x$. 
Theorem

If an output is returned in steps 2 or 11, it is correct. The probability of failure is at most $1/2 + \epsilon$, and for an $n$-bit input $N$ the reduction uses $O(n^3)$ operations in $\mathbb{Z}_N$ plus one call to $B'_5$ with an argument of odd order.