DL-Attacks and their Consequences

The Discrete Log Problem:

\[(DLP) \text{ Given } g \in G \text{ (where } G \text{ is a group), and }
\]
\[y = g^x,
\]
find \(x = DLog_g(y)\), the discrete log of \(y\) to the base \(g\).

General DL-Algorithms/Attacks:

Log-table method: Make a table of all pairs \((x, g^x)\).

Time: \(O(n)\) multiplications; Storage: \(O(n)\) group elements, where \(n = \text{ord}(g)\).

BSGS (= Baby-Step-Giant-Step) - due to Shanks (1970):

Idea: find \(r, s\) such that \(x = r[\sqrt{n}] + s\) by using log tables of size \(\sqrt{n}\).

Time: \(O(\sqrt{n})\) multiplications; Storage: \(O(\sqrt{n})\) elements.

Pollard's \(\rho\) Method - due to Pollard(1978):

Idea: Random walk. Construct a function \(f = f_y : G \rightarrow G\) and express \(y_k = f^k(y_0)\) recursively as \(y_k = g^{x_k} y^{z_k}\) for \(k = 1, 2, \ldots\). Look for coincidences \(y_{k+t_i} = y_k\); these yield relations

\[(x_k - x_{k+t_i}) \equiv x(z_{k+t_i} - z_k) \pmod{n}\]

from which \(x\) can be computed.

Time: \(O(\sqrt{n})\) multiplications; Storage: \(O(1)\) elements.

Pollard's \(\lambda\) (or Kangaroo) Method - due to Pollard(1978):

Exploits the birthday paradox: The footprints of two kangaroos hopping around will eventually cross.
**SPH** - due to Silver, Pohlig, Hellman (1978):

**Idea:** using the Chinese Remainder Theorem, the DLP for $G$ can be reduced to the DLP for its subgroups of prime order $p | n$.

**Time:** $O(\sum_{p|n}(e(p) \log n + \sqrt{p})$ group operations $+ O((\log n)^2)$ bit operations (for CRT).

**Storage:** $O(1)$ elements, if Pollard’s method is used.

**Index Calculus:** - developed out of factorization programs

**Idea:**
1) Set up a factor base $\mathfrak{B}$;
2) Compute the discrete log $x(b) = \text{DLog}_g(b)$ for each $b \in \mathfrak{B}$ by solving a system of linear equations;
3) Find an exponent $z$ such that $yg^z$ is “$\mathfrak{B}$-smooth”, i.e. such that we can determine exponents $e(b)$ with the property that

$$yg^z = \prod_{b \in \mathfrak{B}} b^{e(b)}.$$

Then we have:

$$x = \text{DLog}_g(y) \equiv \sum_{b \in \mathfrak{B}} x(b)e(b) - z \pmod{n}.$$  

**Note:** Although this method works in principle for any (realization of the) group $G$, it is only practical if a suitable factor basis $\mathfrak{B}$ can be found, i.e. one for which the relations leading to the system of linear equations can be found efficiently. Such bases are only known for $G = \mathbb{F}_q^\times$, but not for $G = E(\mathbb{F}_q)$, where $E$ is an elliptic curve.

**Examples:**
1) If $G = \mathbb{F}_p^\times$ ($p$ prime), fix $B << p$ and take

$$\mathfrak{B} = \{\text{primes } q \leq B\}.$$
2) If $G = \mathbb{F}_q^\times$, where $q = p^n$, fix $M < n$ and take

$$\mathfrak{B} = \{p(x) \in \mathbb{F}_p[x] : \deg(p) \leq M, p(x) \text{ irreducible}\}.$$ 

**Time:** If $G = \mathbb{F}_p^\times$, then the above leads to a subexponential algorithm:

$$\text{Time(to solve DLP)} = L_p \left(\frac{1}{2}, c\right)$$

for some $c > 0$, where

$$L_p(r, c) = O \left( e^{c(\log p)^r (\log \log p)^{1-r}} \right).$$

**Consequences:** 1) By SPH, groups of sufficiently large prime order are the safest for cryptosystems based on (DLP).

2) To design a safe cryptosystem based on the (DLP) in $G = \mathbb{F}_q^\times$, one has to take $n$ (and hence $q$) much larger than for a cryptosystem based on the DL-problem in $G = E(\mathbb{F}_q)$.

**Recommendation:** A.K. Lenstra, E.R. Verheul (Sep. 1999) propose the following minimum key sizes (in bits):

<table>
<thead>
<tr>
<th>Year</th>
<th>RSA</th>
<th>SDL</th>
<th>EC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>q</td>
<td>p</td>
<td>wo (w)*</td>
</tr>
<tr>
<td>2000</td>
<td>952</td>
<td>123</td>
<td>132 (132)</td>
</tr>
<tr>
<td>2005</td>
<td>1149</td>
<td>131</td>
<td>139 (147)</td>
</tr>
<tr>
<td>2025</td>
<td>2174</td>
<td>158</td>
<td>169 (202)</td>
</tr>
<tr>
<td>2050</td>
<td>4047</td>
<td>193</td>
<td>206 (272)</td>
</tr>
</tbody>
</table>

Here SDL = subgroup discrete log (use $G \leq F_p^\times$ of size $q$ bits) and EC = elliptic curve discrete log (in $G \leq E(\mathbb{F}_q)$).

(wo/w = without/with cryptographic advances).