Recall – Notations

- n has bit length ℓ_n,
$$2^{\ell_n - 1} \leq n < 2^{\ell_n}.$$
- $a, b \in \mathbb{Z}_n$, in particular, $0 \leq a, b < n$.
- ω: the computer's word size
    - for a 64-bit processor, the *word size* is 64
- Let $\kappa = \lceil \ell_n / \omega \rceil$, i.e. $(\kappa - 1)\omega < \ell_n \leq \kappa\omega$.
- Then ($||$ indicates concatenation, $0 \leq a_i < 2^\omega$)
$$a = a_{\kappa-1} || a_{\kappa-2} || \ldots || a_0,$$
- Note that some a_i might be 0 if the bit length of a is less than ℓ_n. We have
$$a = \sum_{i=0}^{\kappa-1} a_i (2^\omega)^i.$$

Attack on a simple algorithm

Algorithm 3: An algorithm involving computing modular multiplication with Blakely's method.

Input: $n,\ a,\ b,\ c$ // $a, b \in \mathbb{Z}_n$; $c = 0, 1$

Output: $ab \bmod n$ if $c = 1$ and a otherwise

if $c = 1$ **then**
    $R = 0$
    // $\kappa = \lceil \ell_n / \omega \rceil$, where ω is the computer's word size
    for $i = \kappa - 1$, $i >= 0$, $i - -$ **do**
        $R = 2^{\omega} R + a_i b$
        $R = R \bmod n$
    $a = R$
return a