TLDR

: We show a simple technique to blind VDF and timelock outputs. More specifically, let g

be an element in a group of unknown order. We show how to prove knowledge of x = g^{2^t}

without revealing intermediate outputs g^{2^s}

for s \le t

.

Construction

Simply reveal x^3

with a corresponding Wesolowski proof. (Notice the Wesolowski scheme works for arbitrary exponents, not just powers of two.) More concretely, the prover reveals:

- x^3 = g^{3*2^t}
- p = g^{\lfloor{3*2^t/l}\rfloor}

where l

is a 256-bit prime deterministically generated from x^3

The verifier checks p^lg^{3*2^t \mod l} = x^3

.

Blindness argument

(Note

: This is not a formal cryptographic proof. Feedback and corrections welcome.)

Notice g^{2^s}

cannot be extracted from g^{3*2^t}

for s \le t

because that would be taking the 3*2^{t - s}

th root of x^3

and taking roots is assumed to be hard in an RSA group. Also, the sequentiality assumption implies that g^{2^s}

for s > t

cannot be computed from g^{3*2^t}

without computing g^{2^s - 3*2^t}

. At best s = t+2

and g^{2^{t+2} - 3*2^t} = x

must be computed first.

Similar blindness arguments apply to p

. Indeed, the repeated square g^{2^s}

cannot be extracted from p

if 2^s < \lfloor3*2^t/l\rfloor

by the roots assumption. And g^{2^s}

cannot be computed from p

if 2^s \ge \lfloor3*2^t/l\rfloor

without first computing g^{2^s - \lfloor3*2^t/l\rfloor}

which requires (except with negligible probability) essentially as much work as computing x

(because l

has 256 bits and 2^t

has t \gg 256

bits).

Motivation

Blinding of repeated squares was motivated by therefreshed LCS35 puzzle where intermediate outputs are welcome:

CSAIL is also interested in solutions for t = 2 ^ k for 56/2 <= k < 56; these are called “milestone versions of the puzzle”

The above scheme allows to prove that milestone versions of the puzzle were computed without revealing them, and hence protecting against someone building upon them. The scheme may also be helpful in other contexts such as iterated VDFs.