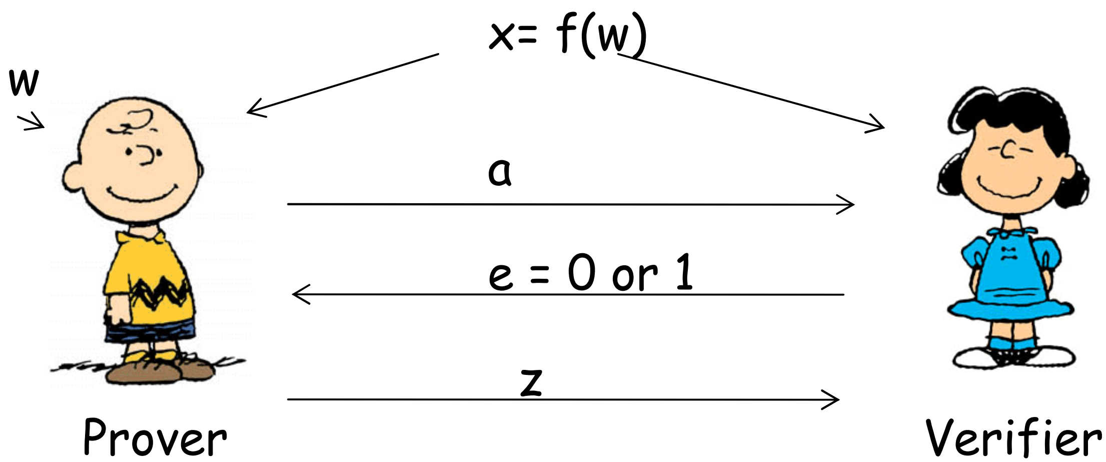
# On the Amortized Complexity of Zero-Knowledge Proofs


Ronald Cramer, CWI

Ivan Damgård, Århus University

# Classic Zero-Knowledge Protocols

- for, e.g., discrete log or quadratic residuosity, are of form




Has error probability ½ . Can be amplified to $2^{-n}$ by iterating n times. Means proof has size O(kn) bits, k size of problem instance.

Some constructions do much better: O(k+n) bits.

- Schnorr: only for groups of public and prime order.
- Guillou-Quisquater: only for q'th roots mod a composite, q a large prime.
- Okamoto-Fujisaki: discrete log in RSA groups, but only under strong RSA assumption and for special moduli.

No better *general* method known for amplifying error.

# Results of this paper

For a large class of problems, we show how to do a zero-knowledge proof for n problem instances simultaneously, such that:

- the complexity per instance proved is $O(n+k)$ bits, and
- the error probability is $2^{-n}$.

Construction is unconditional.

Result works for any function f that has certain homomorphic properties (f is a "zero-knowledge friendly" function):
Given $x_1,\ldots,x_n$, the prover shows he knows $w_1,\ldots,w_n$ such that $f(w_i) = x_i$

Includes

- Discrete log in any group,
- Quadratic residuosity, improves also classic protocol for quadratic non-residues
- Goldwasser-Micali encryptions and similar cryptosystems,
- Integer commitment schemes based on discrete log mod a composite.

# Results cont'd

Result extends to show relations between preimages under f, such as multiplicative relations.

We obtain a Σ–protocol, a 3-move honest verifier zero-knowledge protocol.

Honest-verifier zero-knowledge is enough for many applications.

Upcoming work (Cramer, Damgård and Keller): for same class of problems, can get constant-round proof of knowledge that is zero-knowledge against any verifier, proof has same size as ours up to a constant factor, and properties are unconditional.

**Related Work**

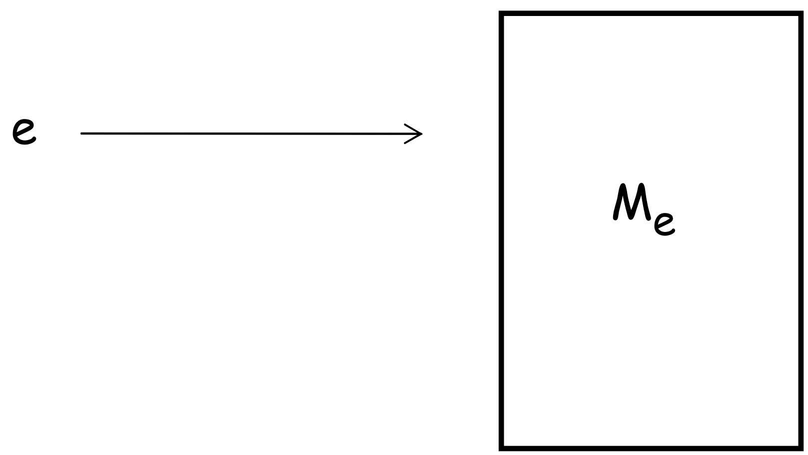
Ishai et al. (STOC 07) have a construction of zero-knowledge protocols from multiparty computation that can give similar complexity as ours for some, but not all problems and requires a complexity assumption.

# The Construction, preliminaries

Let e be an n-bit string.

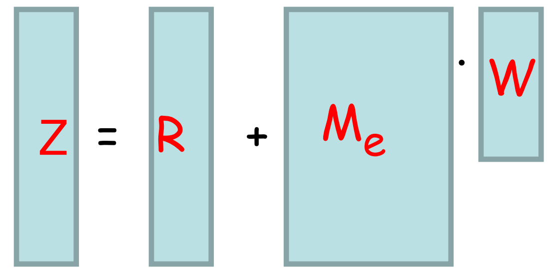
We will need an efficiently computable function: takes e as input and outputs matrix $M_e$, with integer entries.
n columns, m rows. In this example m=2n-1.

Other dimensions possible as well. Details on the function later.



# Idea of construction

- for discrete logarithm in any group




$h_1 = g^{w1}, \ldots, h_n = g^{wn}$

w1,…,wn

$a_1 = g^{r1}, \ldots, a_m = g^{rm}$

$e = e_1, \ldots, e_n$

$z_1, \ldots, z_m$

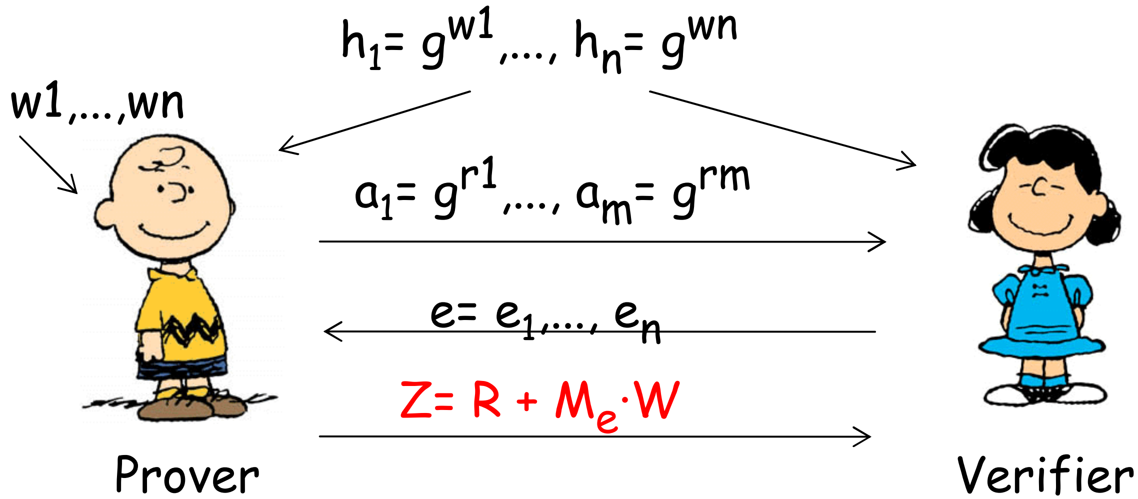
Prover

Verifier

How to compute $z_1, \ldots, z_m$ : Let W, R, Z be columns vectors containing the wi's, ri's and zi's. Then prover sets
$Z = R + M_e \cdot W$

How to check Z is correct: Let $(t_{i1}, \ldots, t_{in})$ be i'th row of $M_e$
must be the case that for i=1 … m:
$g^{zi} = a_i \cdot h_1^{ti1} \cdot \ldots \cdot h_n^{tin} = g^{ri + w1 \cdot ti1 + \ldots + wn \cdot tin}$

# Why is this (honest-verifer) zero-knowledge?




Check that

$g^{zi} = a_i \cdot h_1^{ti1} \cdot \ldots \cdot h_n^{tin}$

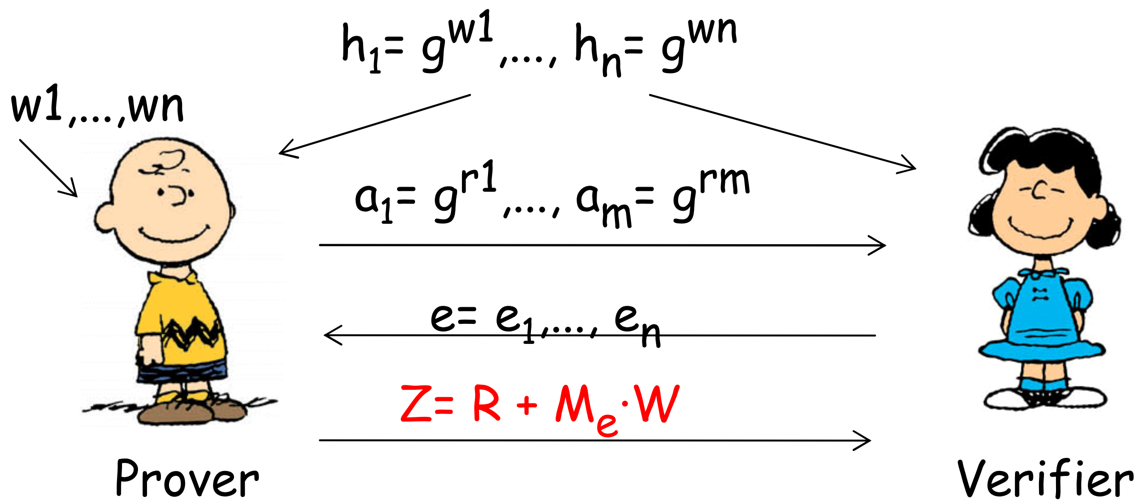
If entries in R chosen uniformly in a large enough interval (compared to entries in $M_e \cdot W$ ) Z will have essentially uniform entries.

Hence, to simulate, choose $z_1, \ldots, z_m$ and e uniformly, compute $M_e$, and compute $a_1, \ldots, a_m$ such that

$$g^{zi} = a_i \cdot h_1^{ti1} \cdot \ldots \cdot h_n^{tin}$$

is true.

# Why is this sound?



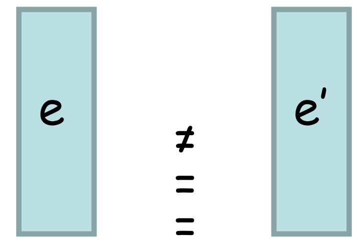

We show that if, after sending first message, the prover can answer two different challenges e,e', then he could compute $w_1, \ldots, w_n$, so error probability is $2^{-n}$.

Intuition on this: if prover can produce
$Z = R + M_e \cdot W$ and $Z' = R + M_{e'} \cdot W$, then he can also compute
$Z - Z' = (M_e - M_{e'})W$

So if we can construct $M_e$ from e such that this equation can always be solved for W, we are done.

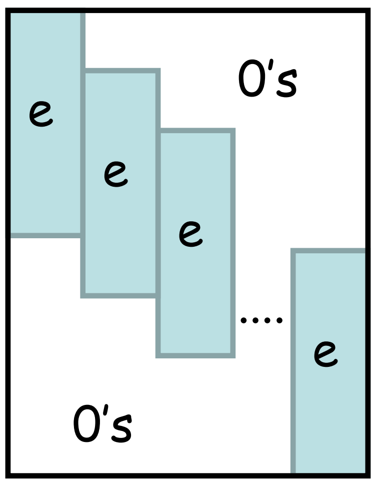
# Construction of $M_e$ from e

Write e as an n-bit column vector




Form the matrix..

$M_e$=



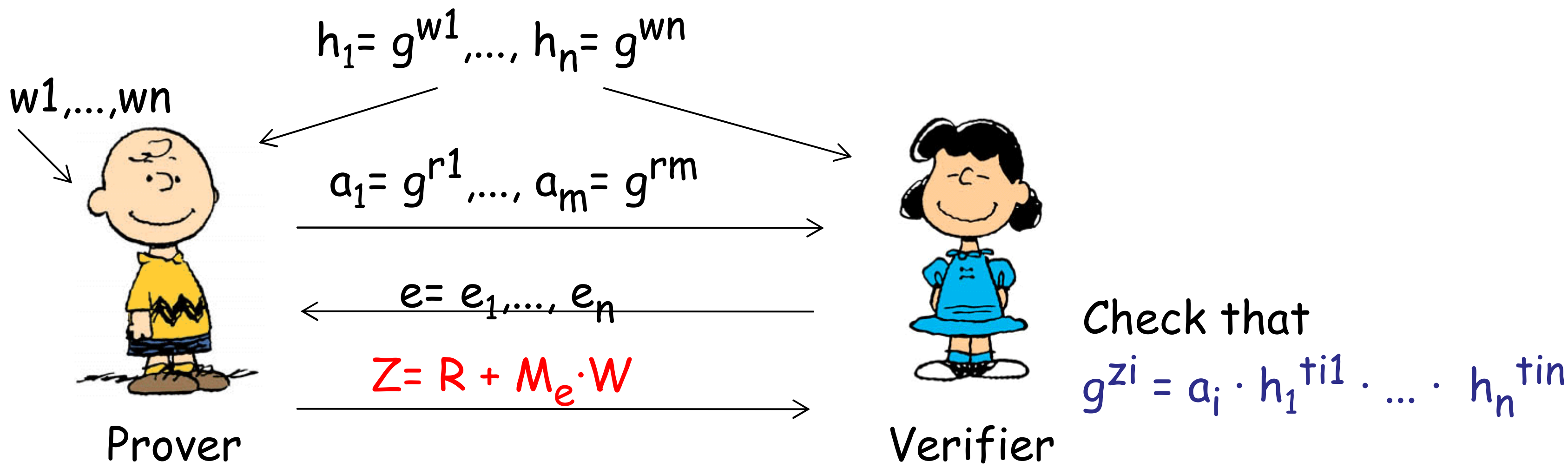

We will get m= 2n-1 rows.

Observation: any difference $M_e - M_{e'}$ is an upper triangular matrix with either +1 or -1 on the diagonal

Why? focus on "lowest" position where e is different from e'.

This implies $M_e - M_{e'}$ is invertible.

# Complexity




**Communication**

Per instance proved, we have sent m/n group elements and numbers.

m/n< 2, so same complexity per instance as Schnorr up to a factor 2.

**Computation**

Entries in $M_e$ are 0, 1, or -1, so computations involving $M_e$ are dominated by the exponentiations. Hence also computation per instance same as Schnorr up to a factor 2.

# In general..

The homomorphic property of the function $w \rightarrow g^w$ is what makes this work.
Many other functions are fine as well, see paper for general framework.

**Examples**:
Not limited to one base, can do proofs of knowledge for $(w,s) \rightarrow g^w h^s$.

Covers several known cryptosystems (Goldwasser-Micali, Groth, Damgård-Geisler-Krøigaard)

- And commitment schemes for committing to integers (Fujisaki Okamoto)

# More Examples

The function $w \rightarrow w^2 \bmod N$
Here special purpose construction of $M_e$ makes it even more efficent:

Consider that n-bit string e can be thought of as an element in $GF(2^n)$.

$GF(2^n)$ is a vector space over GF(2), and multiplication by e is a linear mapping. So fix some basis and let $M_e$ be the matrix of this mapping.

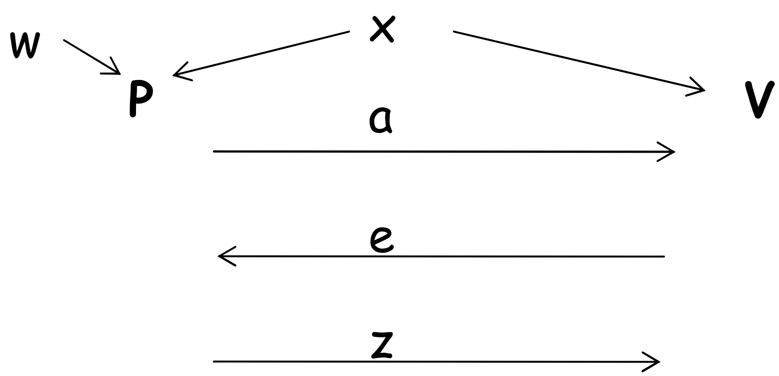
Then any $M_e - M_{e'}$ is invertible because it corresponds to multiplication by $e-e' \neq 0$.

Leads to protocol for proving you know square roots mod N of x1,…,xn. Size of proof per instance is *exactly* equal to one run of the classic GMR protocol.

# Also in Paper..

Interesting connection between construction of $M_e$ and black-box secret sharing.

Most known efficent protocols (Schnorr, G-Q, ours) can be thought of as being based on a 2 out of T secret sharing scheme, for very large T:




w: secret, x: commitment to secret

P commits to randomness for s.s.

V asks for e'th share of secret

Prover reveals requested share,
V checks share is correct

Zero-knowledge because one share does reveal the secret.
Sound because given two correct shares, secret can be computed.