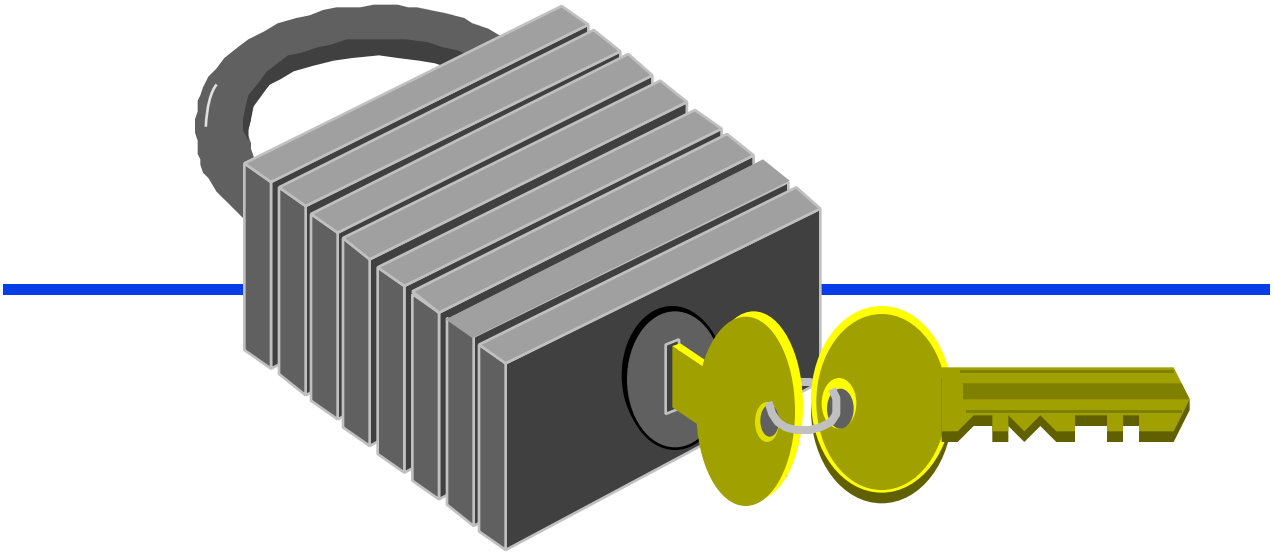


Public-Key Cryptosystems from Lattice Problems

Oded Goldreich

Shafi Goldwasser

Shai Halevi

Motivation

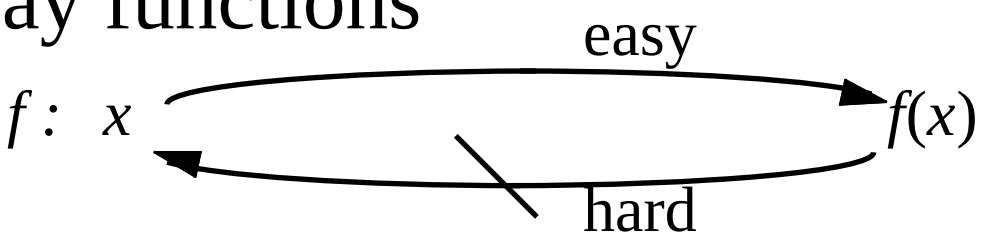
More hard problems in cryptography

- Most public-key systems are based on either factoring or discrete-log
  - These problems seem to be related
- This is potentially dangerous

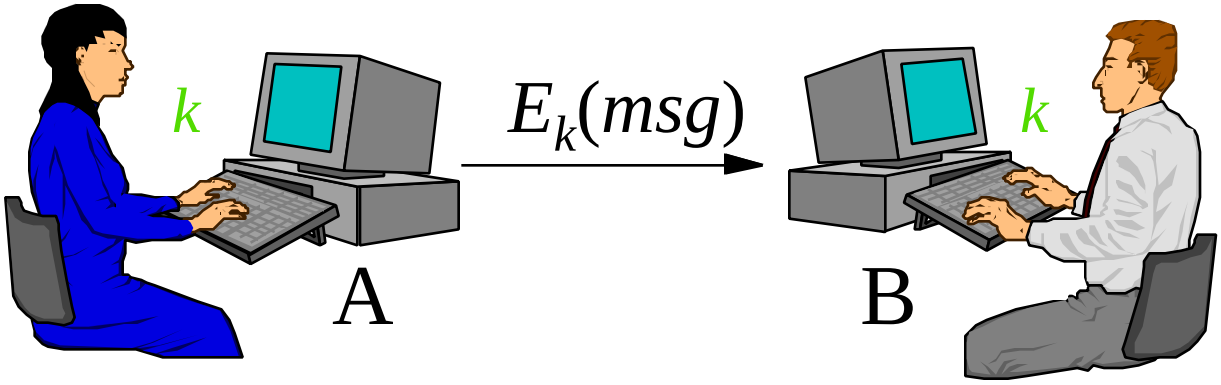
Don't put all the cryptographic eggs in one basket

One-Way vs. Trapdoor Functions

One-way functions

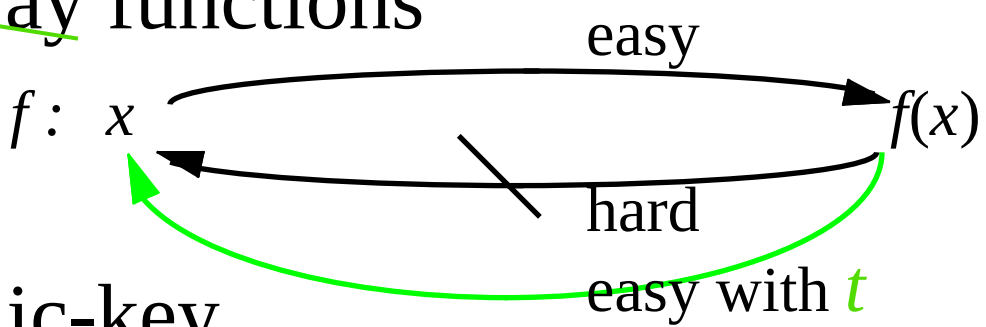


- Secret-key encryption

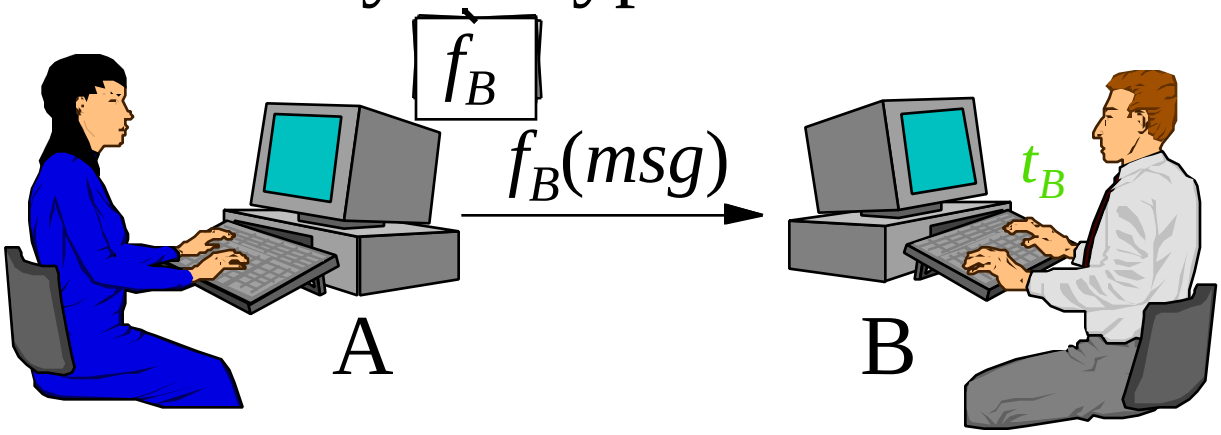


- [Ajtai 96] Based on the "shortest non-zero lattice vector" problem
  - [GGH 96a] Collision-intractable hashing

Trapdoor
~~One-way~~ functions



Public-key
- ~~Secret-key~~ encryption



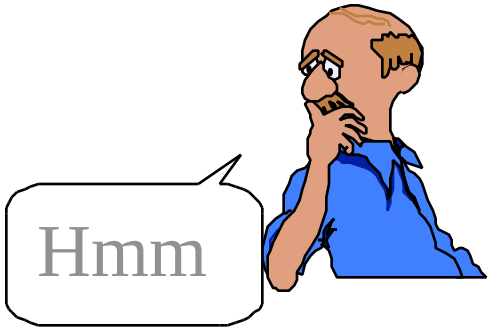
No prior agreement is required

- Much harder to come by
  - Need “strong mathematical properties”

Goal: Trapdoor Functions

Where to find such properties ?

- ~~Number-theory~~
  - Large body of knowledge
- Geometry
  - ditto
- Many details



Formally - 3 algorithms:

- Key generation $(f, t) \leftarrow G(random)$
- Evaluation $c=f(x) \leftarrow E(f, x)$
- Inversion $x=f^{-1}(c) \leftarrow I(f, c, t)$

Overview

A “crash course” on lattices

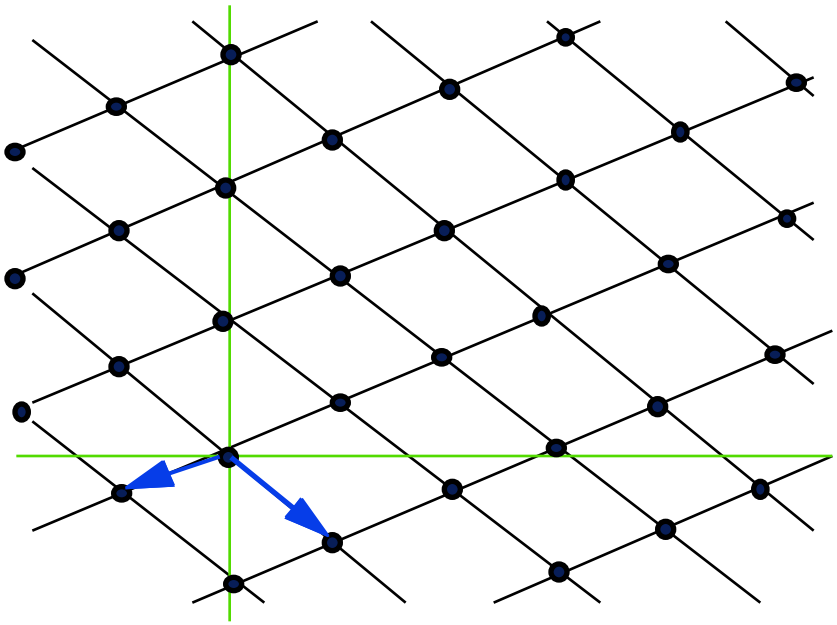
- Lattices and bases
- Lattice-reduction problems
- The LLL algorithm

A trapdoor function

- The construction
- Security

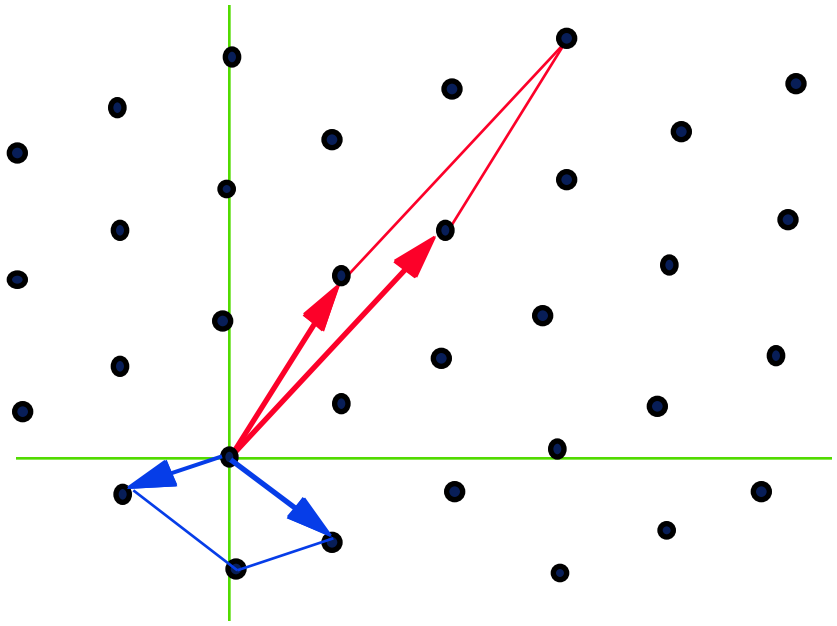
Lattices

- $v_1 \ldots v_n$ - linearly independent in R^n
  $B = (v_1 \ldots v_n)$ - invertible matrix
- $L(B) = \{\ Bx\ :\ x$ is an integer vector $\}$
- B is a basis of $L(B)$
  - Tiling of R^n with the parallelepiped spanned by B, lattice points are vertices



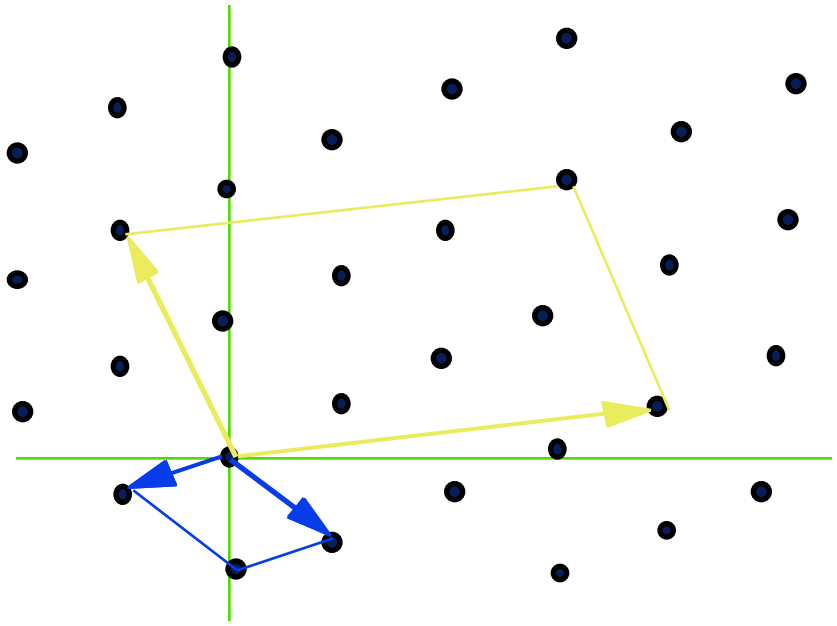
Bases

- A lattice L has many bases



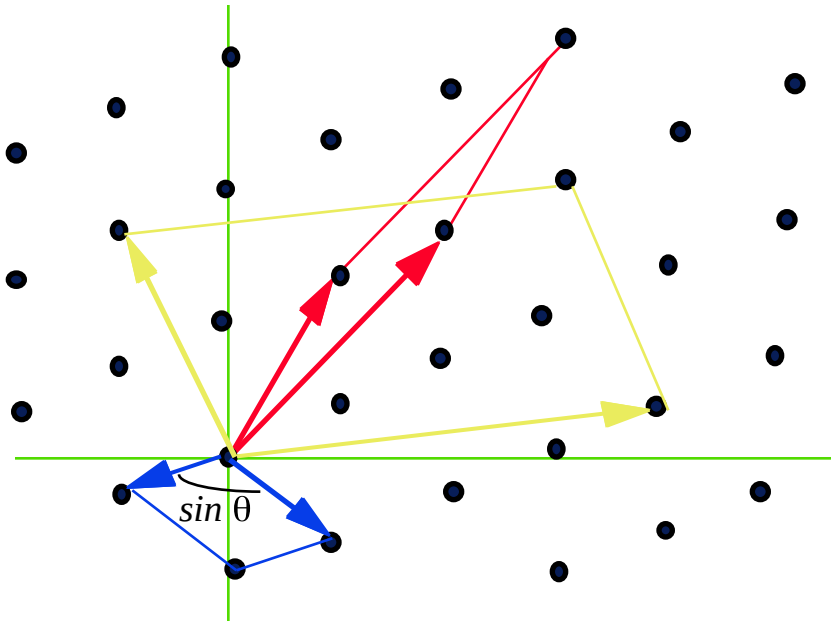
- Clearly, red-vectors in blue lattice
- Two ways to see other direction
  - left-blue = right-red − left-red
    right-blue = −left-red − 2 left-blue
  - Observe that red parallelepiped has the same volume as blue one => to fill R^2 you must place one at each lattice point

Bases (2)



- Not everything is a basis
- Yellow vectors span a sub-lattice
  - The yellow parallelepiped is too large. Can't fit one for each lattice-point

Size of Bases



- If $L(B) = L(B')$ then $det(B) = det(B')$
- small vectors <=> large angles

We like bases with “small vectors”

- ~~Orthogonality-defect~~ Size of B = $(v_1 \ldots v_n)$

$$\text{size}(B) = \frac{\prod \|v_i\|}{det(B)}$$

- $det(B)$ is the same for all bases of L
- size is small => each v_i is small

Lattice Problems

Closest vector problem(CVP)

- Given B and c R^n, find the closest point to c in $L(B)$
  - NP-hard in any norm

Smallest basis problem (SBP)

- Given B, find B' of minimum size, so that $L(B) = L(B')$

No polynomial-time algorithm is known for any of them

Lattice-Reduction Algorithms (approx.)

The LLL algorithm and its variants

- Run in polynomial time
  - Feasible up to dimensions $n \approx$ 200-300 (?)
- Approximate SBP (input B, output B')
  - Exponential ratio (in n) in the worst case
  - Independent of the size of the numbers
  - "Typically" work much better
- Imply approximation for CVP
  - Exponential ratio (in n) in the worst case
  - Smaller B' => better approximation

Summary

- Lattices
- Bases
- Size of bases
- Computational problems
  - CVP
  - SBP
- Approximation algorithms

Moving on . . .

Our Construction

Idea 1: $p \quad L(B), \; e \quad R^n$ is "short"

- Easy to compute $c = p + e$
- Hard to find p from B and c (CVP)
- ⇨ Adding a small error is "one way"

Idea 2: R "smaller" than B

- R yields better CVP-approximation than B
- ⇨ B is public, R is the trapdoor
  - (B, R) ← $G(random)$
  - $f = B$
  - $t = R$

But . . .

Generating (B, R)

Finding R from B is hard (SBP)

- How to get the trapdoor ?
- First pick R, then compute B
  - E.g., $B_1 = R_1 + 150\ R_2 - 342\ R_3 \ldots$
- B is sure to be large

Why should R be small ?

- Recall: small vectors <=> large angles
- Pick the vectors in R with large angles
- R is sure to be small

(Randomized) Trapdoor Function

Key generation (in dimension n)

- Trapdoor is a "small basis" R
- Function-description is (B,σ)
  - B is a "large basis", σ is a real number

Computing the function

- Input is x (integer vector), and (B,σ)
- Pick a random error vector e
  - Each entry in e is chosen as $\pm\sigma$
- Compute $c = Bx + e$

lattice point
call it p

Inverting the function

- Input is (B, c, R)

❶ Use R to find the lattice point p

  – Using a CVP approximation technique

❷ Compute $x = B^{-1}p$

Round-off Technique [Babai 86]

- Input is R and $c = p + e$ $(e_i = \pm\sigma)$

① Represent c as a linear combination of R's columns

② Round coefficients to nearest integers

③ The integer combination is a lattice point, $p' = R\,[R^{-1}c]$

It Works

Thm: For any $\varepsilon \geq 0$, can compute σ such that $\Pr_e[p' \neq p] \leq \varepsilon$

- "Typical values" ($n = 140$)
  R_{ij} $\{-4, \ldots, +4\}$, $\varepsilon = 10^{-4}$, $\sigma = 2.7$

Details

- Picking R
- Computing B from R
- Constructing a (deterministic) function
- Using the trapdoor-function for encryption

Security

Two main threats

- Recovering the trapdoor:
  Finding R from B is hard (SBP)
- Bypassing the trapdoor:
  Finding p from (B ,c) is hard (CVP)

Solving SBP, CVP is hard in general

- An attack must exploit the particular instances that come out of our system

Security Conjecture: Hard to solve SBP and CVP instances from the distribution induced by our system

Substantiating the conjecture

- Try many attacks
- Evaluate their work-load

Various Attacks

Attack #0: Exhaustive search for e

- $e_i = \pm\sigma$ => takes 2^n trials

Attack #1: Use B for round-off

① "LLL Reduce B", try $p' = B[B^{-1}c]$

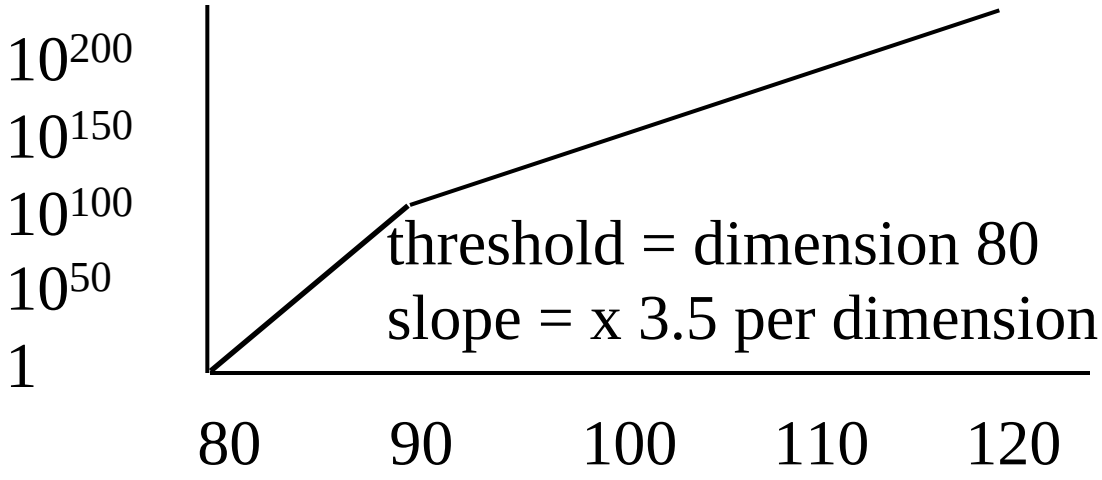
② Use p as starting-point for search

  – If, e.g., $(B^{-1}c)_i = 4.6$, try both 4 and 5

- Analysis: Work-load ≈ $size((B^{-1})^t)$

  – What $size((B^{-1})^t)$ comes out of LLL ?

- Extensive testing: Evaluate $size((B^{-1})^t)$



Other Attacks

Work in progress ...

- Better approximations for CVP
- Better variants of LLL (Schnorr)
- . . .
- ⇨ Push the threshold up, a more moderate slope (but still exponential)

Tentative conclusion:

- Secure “in asymptotics”
- “Practical security” at $n \approx 300$ (?)

Digital Signatures

Idea: Signature on a point in R^n is a "fairly close" lattice point

- Secret-key: A "small basis" R
- Public-key: A "large basis" B and a threshold τ
- Message: An arbitrary point $x \quad R^n$
- Signature: A point $p \quad L(B), \|x-p\| < \tau$

Nearby points have same signature

- Bad if they represent different data
- Good if they represent the same data
- ⇨ May be useful for signing analog data
  - E.g., signing a FAX, or a musical piece
  - Small changes don't invalidate signature

Conclusions

- A new proposal for trapdoor function
- Based on different “hard problems”
- Fast evaluation/Inversion
  - matrix-vector multiplication
- Implies public-key encryption
- Implies digital signatures

Future Work

Security evaluation of [GGH96]

- More tests
- “Proof of security” (?)
- Connection with error-correction coding

More efficient schemes

- Faster encryption/decryption
- Smaller keys

Other useful hard-problems

- Geometry over finite fields/rings/groups
- Non-Euclidean geometry

Schemes “as secure as NP vs. BPP”(?)