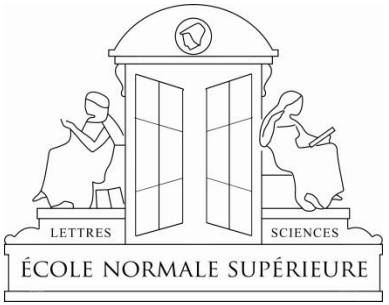
The LPN Problem in Cryptography

Vadim Lyubashevsky

INRIA / ENS, Paris

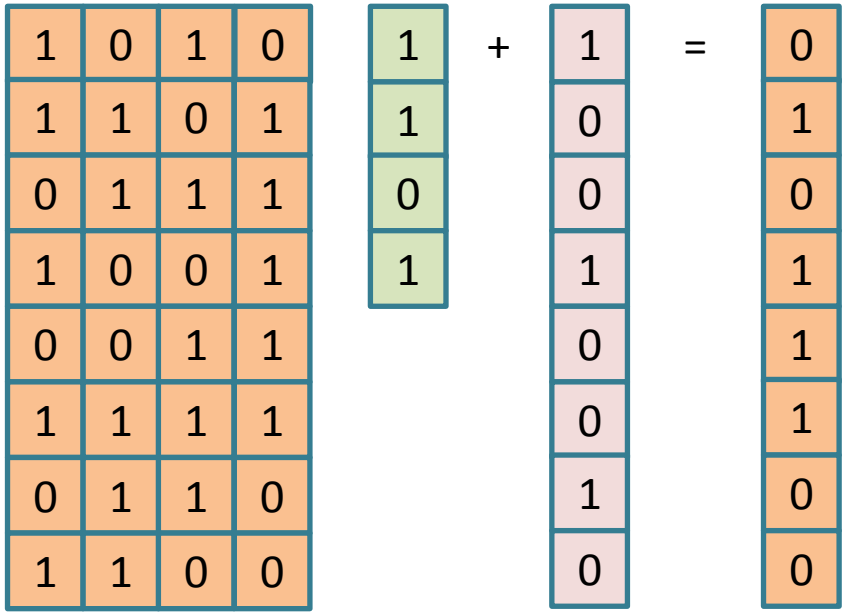


Learning Parity with Noise (LPN)

We have access to an oracle who has a secret **s** in $\mathbf{Z}_2^n$

On every query, the oracle:

1. Picks $\mathbf{r} \leftarrow \mathbf{Z}_2^n$
2. Picks a `noise' e $\leftarrow \beta_{1/4}$ (i.e. e= 0 w.p. ¾ and 1 w.p ¼)
3. Outputs (**r**, t=<**r**,**s**> + e)



The goal: Find **s**

Algorithms to Solve LPN

Important Parameters

1. Dimension
2. Noise rate
3. # of samples

Straightforward algorithm:

check which **s** is the best fit

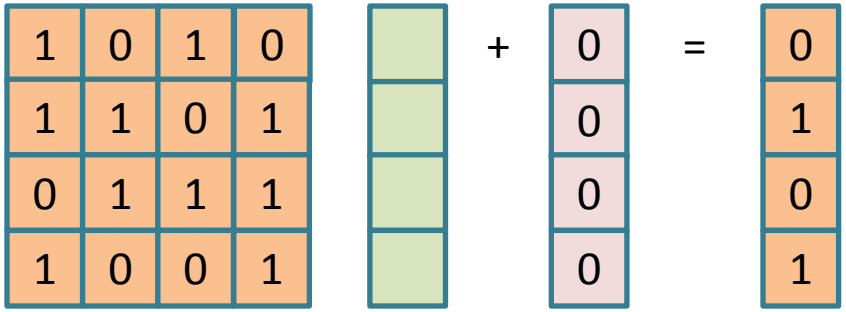
2^n time, any noise rate, minimum samples

Better than 2^n ?

If the noise is chosen from β_τ:

get n samples

with probability $(1-\tau)^n$, the noise is all 0's



use Gaussian elimination to solve for **s**

get more samples to check **s**

Solve LPN in time ~ $\exp(\tau n)$

Better than 2^n for Constant Noise Rate [BKW '00, Wag '02]

Find **s** one coefficient at a time

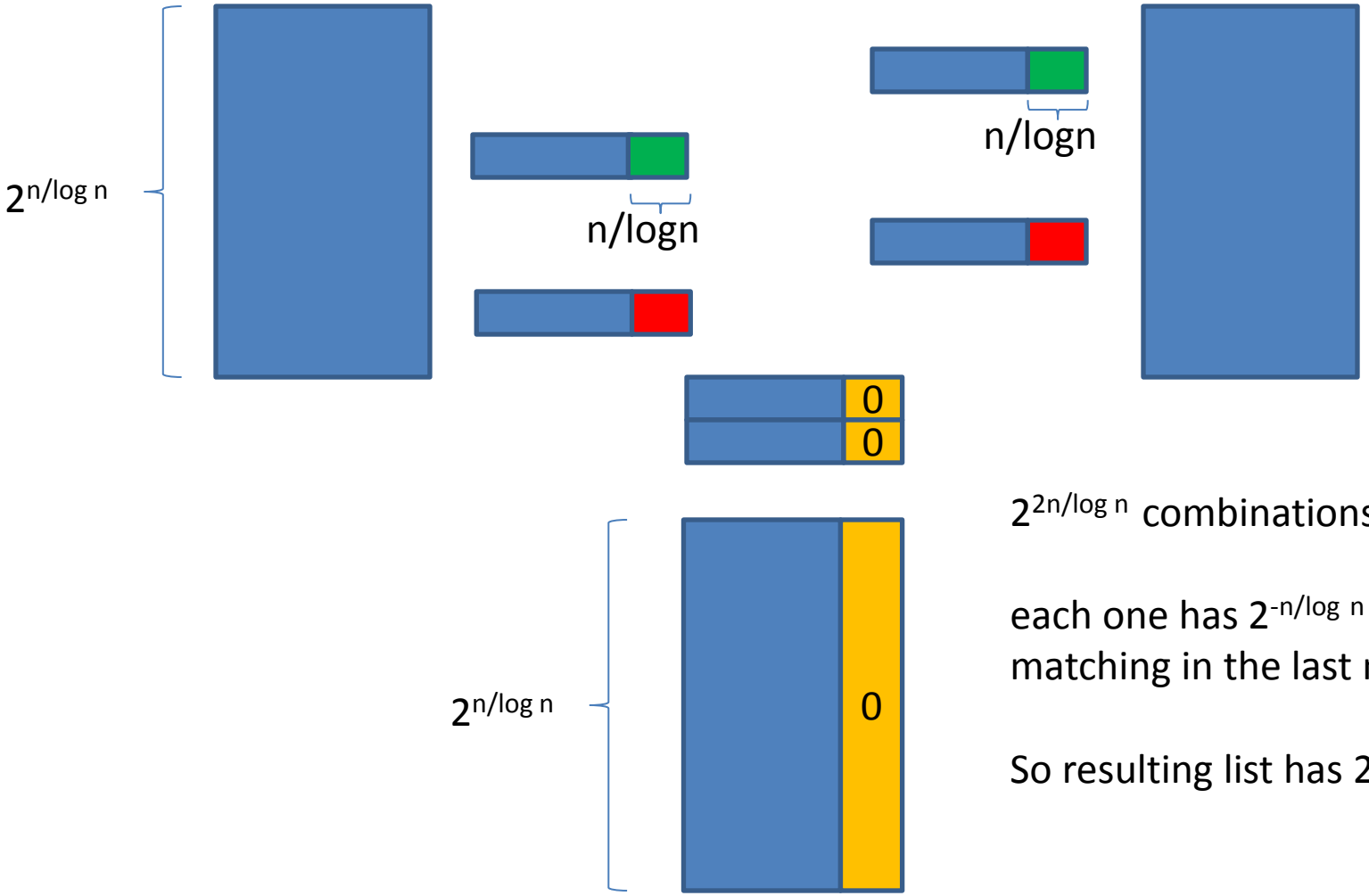
Main idea (for finding 1st coefficient of **s**):

1. Find a linear combination of m samples such that
   $\mathbf{a}=\mathbf{a_1}+\ldots+\mathbf{a_m} = (1000 \ldots 00)$
2. if $\langle \mathbf{a_i},\mathbf{s} \rangle = b_i$ with prob. $1-\tau$, then
   $\mathbf{s}^{(1)} = \langle \mathbf{a},\mathbf{s} \rangle = b_1+ \ldots + b_m$ with prob. $\frac{1}{2} + \frac{1}{2}(1-2\tau)^m$
3. repeat step 1 ~ $(1-2\tau)^{-m}$ times to find many such **a** and determine $\mathbf{s}^{(1)}$ by majority vote

If m=n, can do step 1 in poly(n) time by Gaussian elimination

If $m= n^{\Omega(1)}$, can do step 1 in in time ~ $2^{O(n/\log n)}$

List Merging

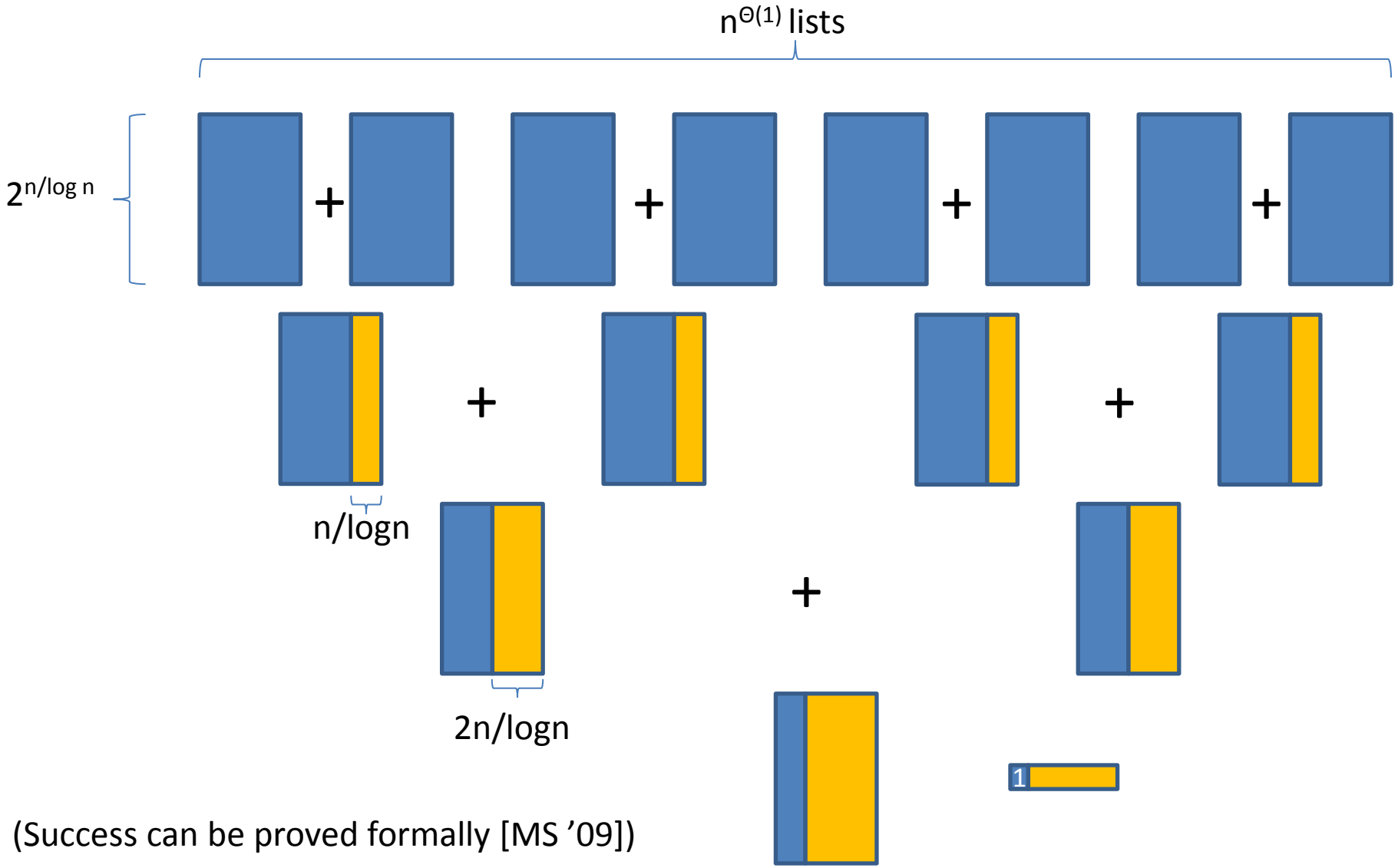


$2^{2n/\log n}$ combinations

each one has $2^{-n/\log n}$ chance of matching in the last n/log n spots

So resulting list has $2^{n/\log n}$ elements

Building a Tree



(Success can be proved formally [MS '09])

Limited # of Samples

- # samples = unlimited → best algorithm $2^{O(n/\log n)}$ time
- # samples = O(n) → best algorithm $2^{\Omega(n)}$ time
- # samples = $\Omega(n^c)$ for c>1
  - Can solve the problem in time $2^{n/\log\log n}$ [Lyu '05]
  - Main idea:
    - Combine n/log n samples at a time to create $2^{\Omega(n/\log\log n)}$ random-looking samples with bigger noise
    - Can prove that the resulting samples are uniform and independent and the noise is not too big (although it is bigger)
    - Use previous algorithm with O(log n/loglog n) lists of $2^{\Omega(n/\log\log n)}$ elements each.

Open Problems

1. Improve the $2^{O(n/\log n)}$ time algorithm
2. Improve the $2^{O(n/\log\log n)}$ time algorithm for polynomially-many samples
3. Improve the $2^{O(n)}$ time algorithm for O(n)-many samples
4. Improve the $2^{O(\tau n)}$ time algorithm when $\tau = o(1/\log n)$
5. Improve practical attacks (i.e. the constants in the exponents)

Cryptography from LPN

1. Public-Key Encryption
2. Authentication Schemes
3. MACs

Equivalent Versions of LPN

We have access to an oracle who has a secret $\mathbf{s}$ in $\mathbf{Z}_2^n$

On every query, the oracle:

1. Picks $\mathbf{r} \leftarrow \mathbf{Z}_2^n$
2. Picks a `noise' e $\leftarrow \beta_{1/4}$ (i.e. e= 0 w.p. ¾ and 1 w.p ¼)
3. Outputs ($\mathbf{r}$, t=<$\mathbf{r}$,$\mathbf{s}$> + e)

We have access to an oracle who has a secret $\mathbf{S}$ in $\mathbf{Z}_2^{n \times n}$

On every query, the oracle:

1. Picks $\mathbf{r} \leftarrow \mathbf{Z}_2^n$
2. Picks a `noise' $\mathbf{e} \leftarrow \beta_{1/4}^n$ (i.e. e= 0 w.p. ¾ and 1 w.p ¼)
3. Outputs ($\mathbf{r}$, $\mathbf{t}=\mathbf{Sr} + \mathbf{e}$)

Equivalence by the hybrid argument

Equivalent Versions of LPN

We have access to an oracle who has a secret $\mathbf{s}$ in $\mathbf{Z}_2^n$

On every query, the oracle:

1. Picks $\mathbf{r} \leftarrow \mathbf{Z}_2^n$
2. Picks a `noise' e $\leftarrow \beta_{1/4}$ (i.e. e= 0 w.p. ¾ and 1 w.p ¼)
3. Outputs ($\mathbf{r}$, t=<$\mathbf{r}$,$\mathbf{s}$> + e)

We have access to an oracle who has a secret $\mathbf{s} \leftarrow \beta_{1/4}^n$

On every query, the oracle:

1. Picks $\mathbf{r} \leftarrow \mathbf{Z}_2^n$
2. Picks a `noise' e $\leftarrow \beta_{1/4}$ (i.e. e= 0 w.p. ¾ and 1 w.p ¼)
3. Outputs ($\mathbf{r}$, t=<$\mathbf{r}$,$\mathbf{s}$> + e)

Choosing **s** same as **e** [ACPS '09, Kir '11]

$$\mathbf{A_1 s + e_1 = b_1} \rightarrow \mathbf{s = A_1^{-1}(b_1 + e_1)}$$

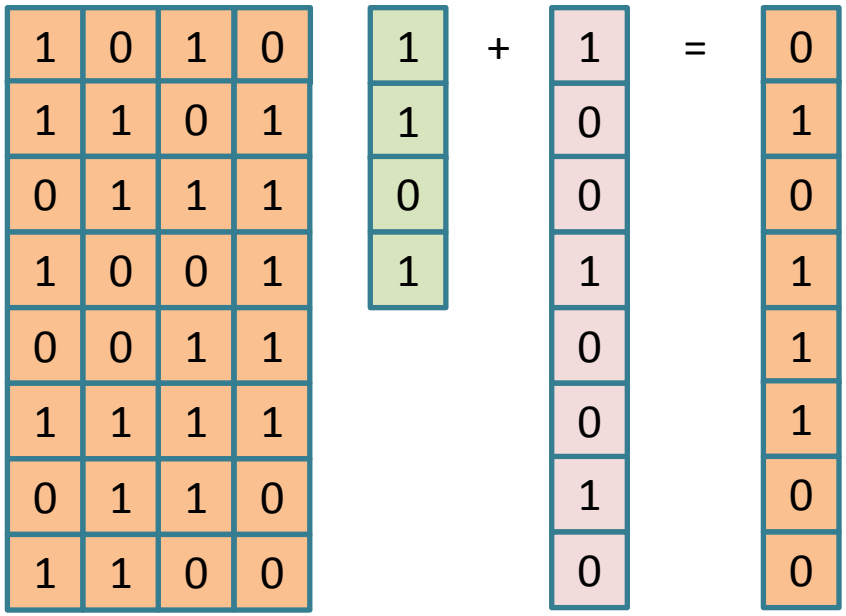
$$\mathbf{A_2 s + e_2 = b_2} \rightarrow \mathbf{A_2 A_1^{-1} e_1 + e_2 = b_2 + A_2 A_1^{-1} b_1}$$

$$\mathbf{A_3 s + e_3 = b_3} \rightarrow \mathbf{A_3 A_1^{-1} e_1 + e_3 = b_3 + A_3 A_1^{-1} b_1}$$

$$\mathbf{A_4 s + e_4 = b_4} \rightarrow \mathbf{A_4 A_1^{-1} e_1 + e_4 = b_4 + A_4 A_1^{-1} b_1}$$

Change input $(\mathbf{A_i, b_i}) \rightarrow (\mathbf{A_i A_1^{-1}, b_i + A_i A_1^{-1} b_1})$

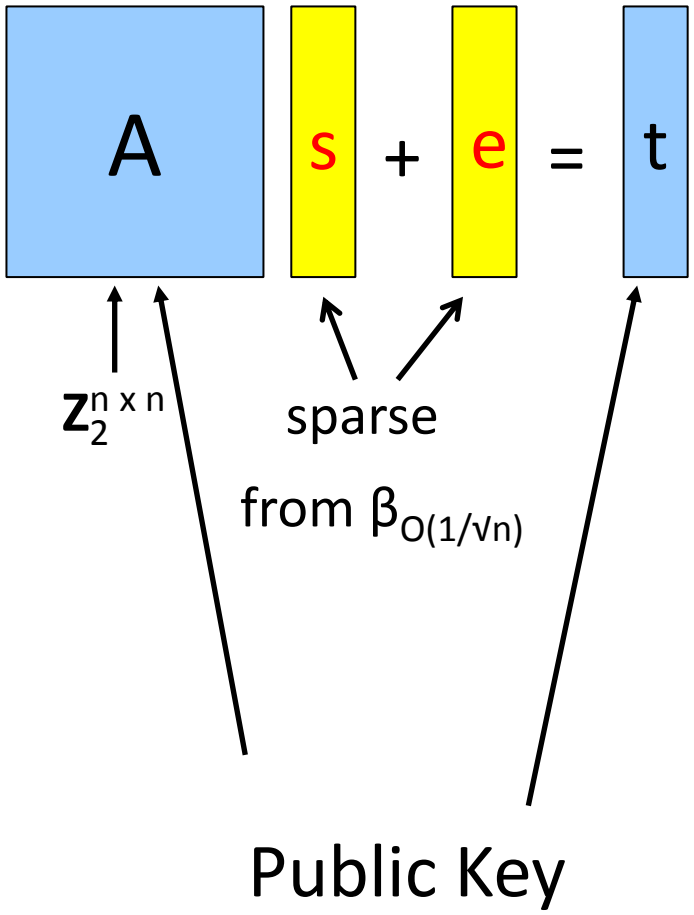
Decision LPN

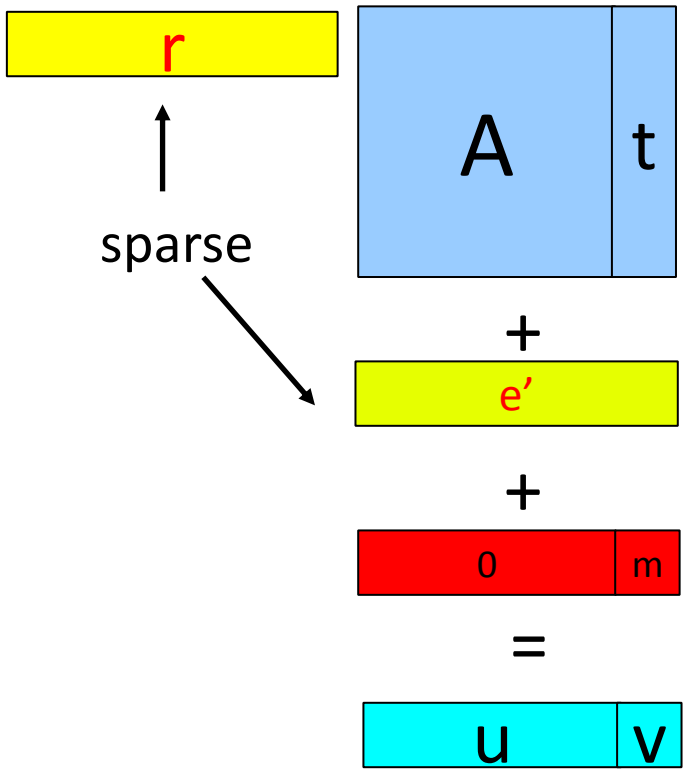


can’t distinguish from uniform

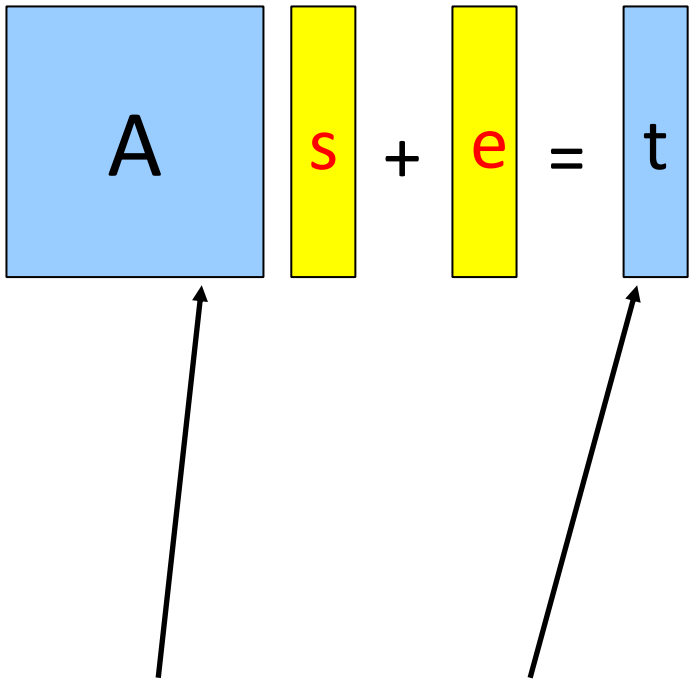
Thm [BFKL ‘93]: Decision-LPN is as hard as LPN

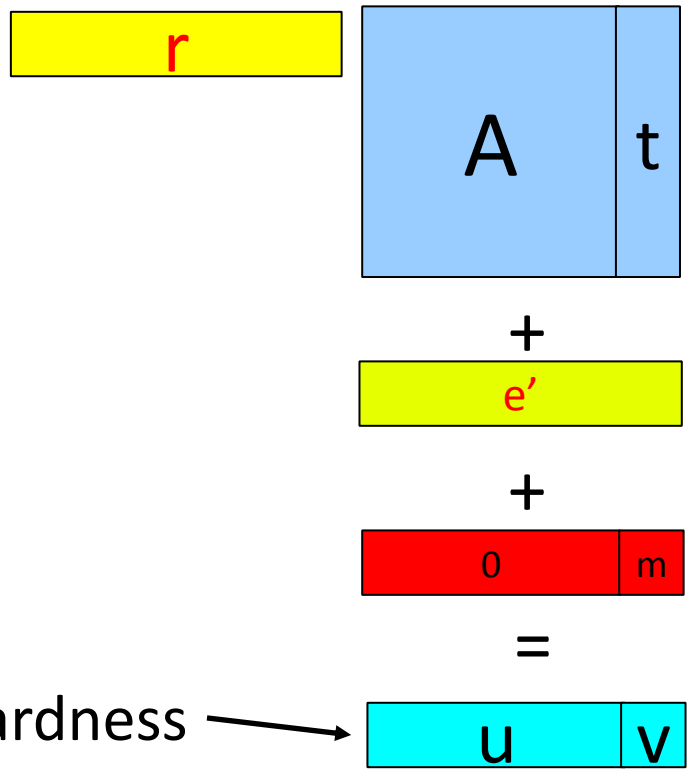
Encryption Scheme (based on [Ale ‘03])





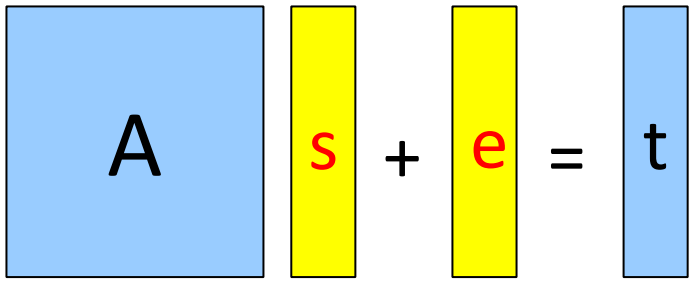
Encryption Scheme

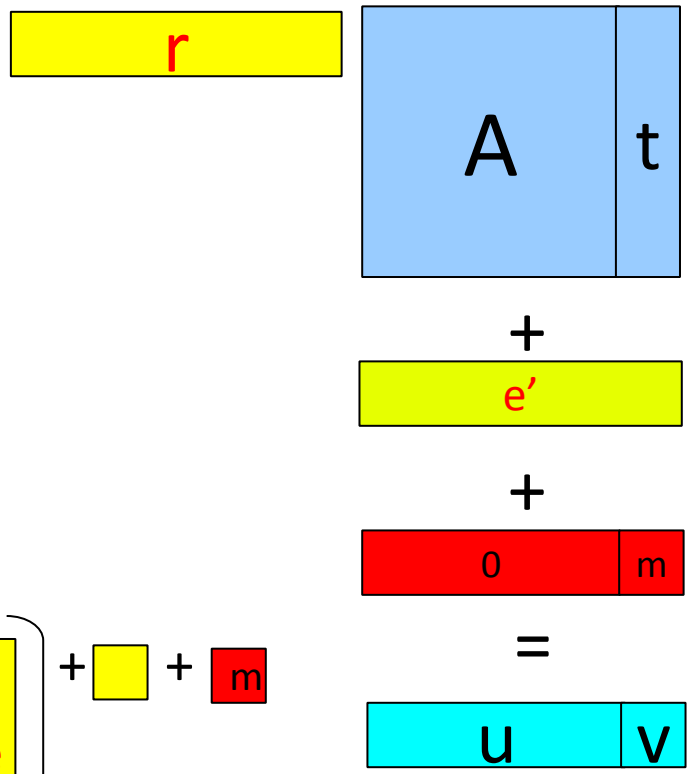


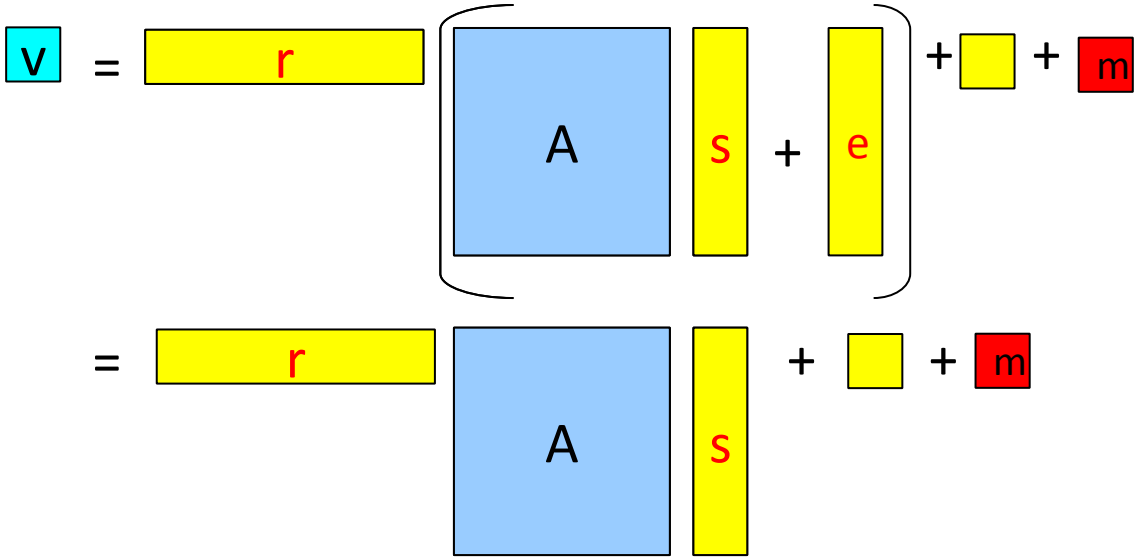


Is pseudo-random based on the hardness of LPN

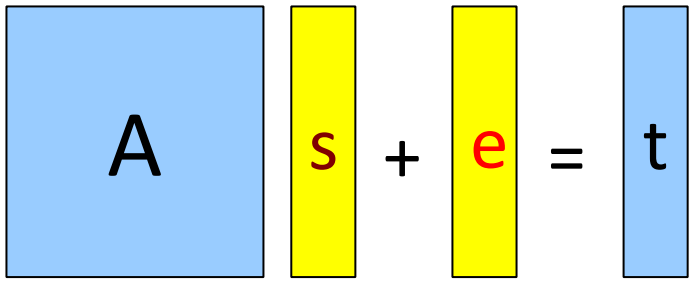
Encryption Scheme

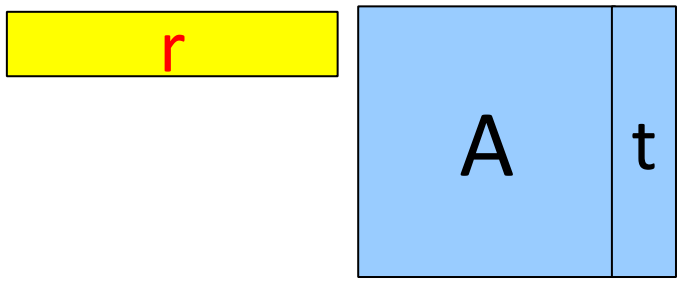


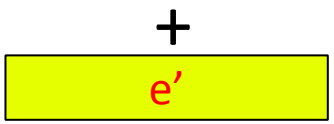


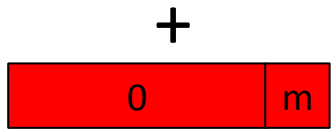


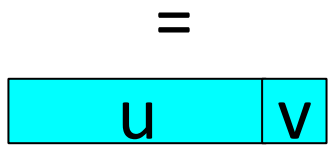
Encryption Scheme

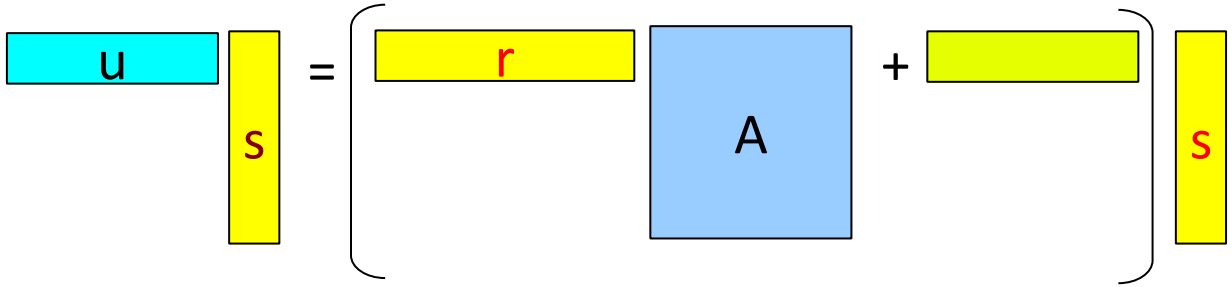


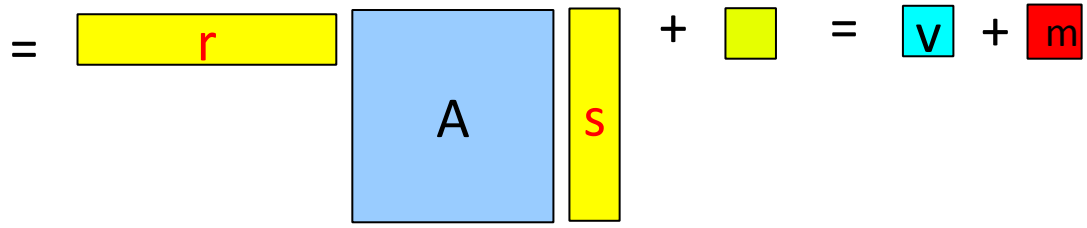






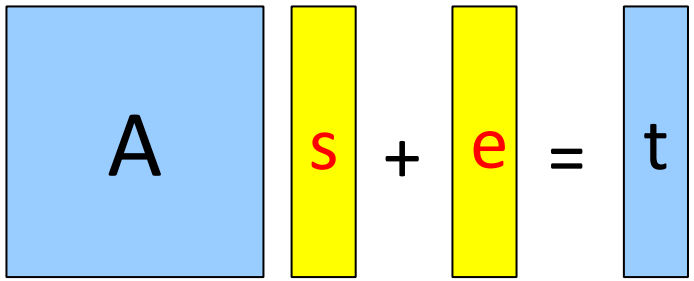


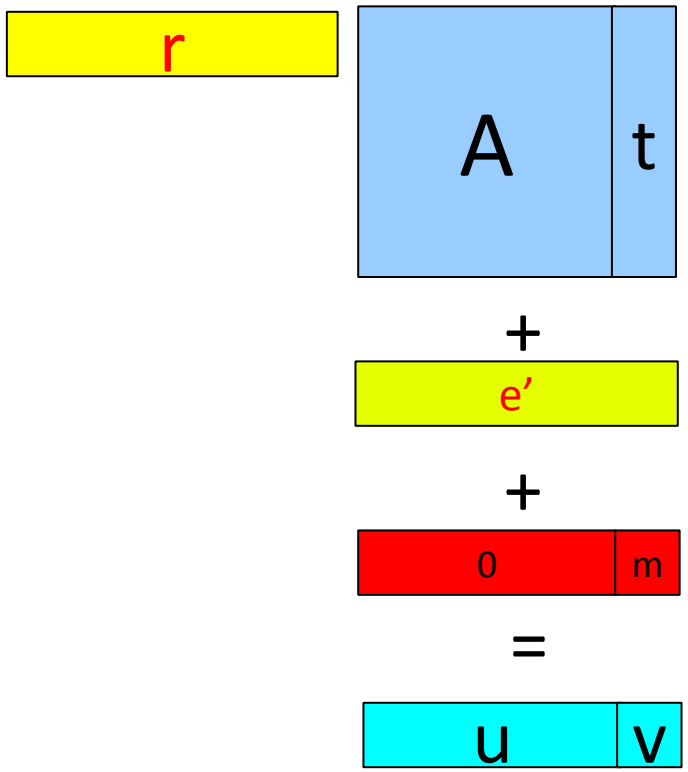


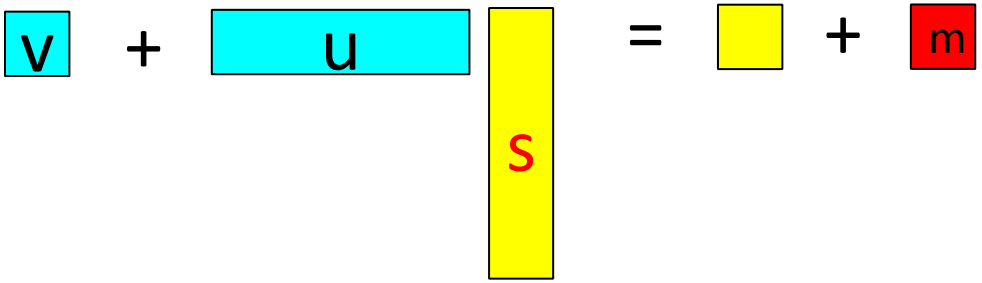


w.p. > 1/2

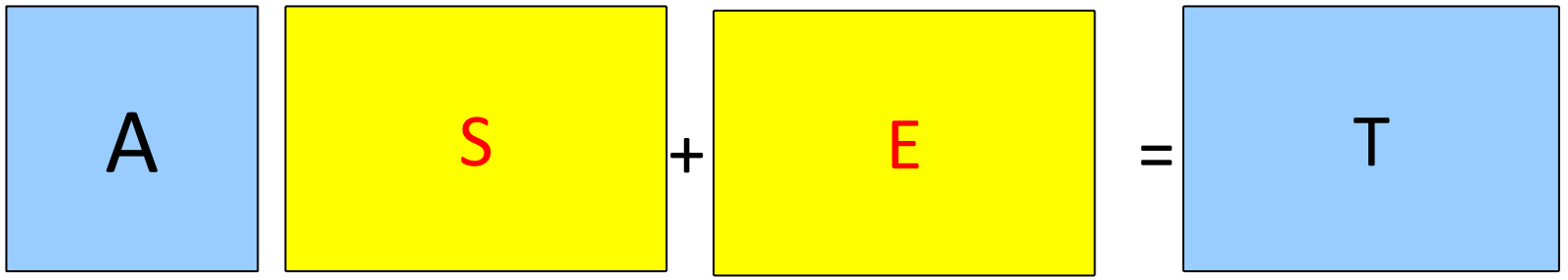
Encryption Scheme

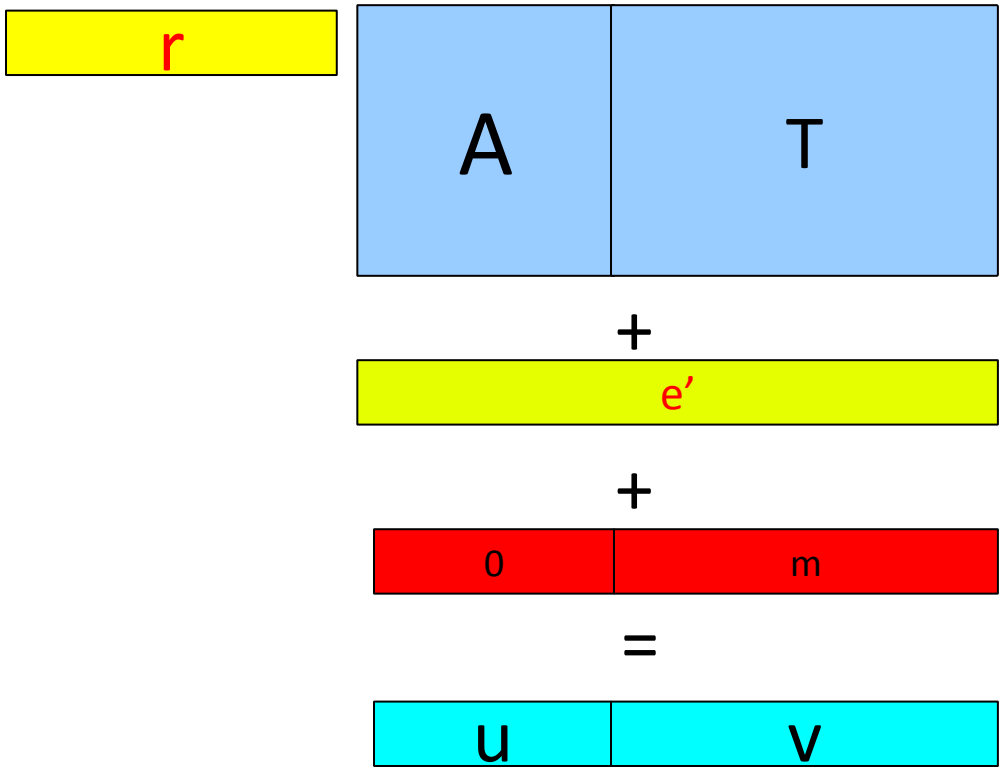






Multi-Bit Encryption Scheme





Open Problems

1. Increase the error in PKE from $O(1/\sqrt{n})$

2. Construct other public key primitives
   - Identity-Based Encryption
   - Efficient Digital Signatures

3. Build a Collision-Resistant Hash Function from LPN

Authentication Protocols

- First protocol (passively secure) [HB '01]
- Since then, there were **many** proposals of more secure variants (e.g. HB^{+} HB^{++}, $HB^{\#}$, etc.)
- Many proposals without security proofs
- Many attacks
- A very confusing state of affairs

- In the past 3 years things got sorted out
- Some new interesting questions opened up

An Abstraction: Weak Pseudo-Random Functions

A Family of functions F: D → R is a weak-PRF family if:

for a random f ← F and $d_1, d_2, \ldots \leftarrow D$

$(d_1, f(d_1)), (d_2, f(d_2)), \ldots$ is indistinguishable from

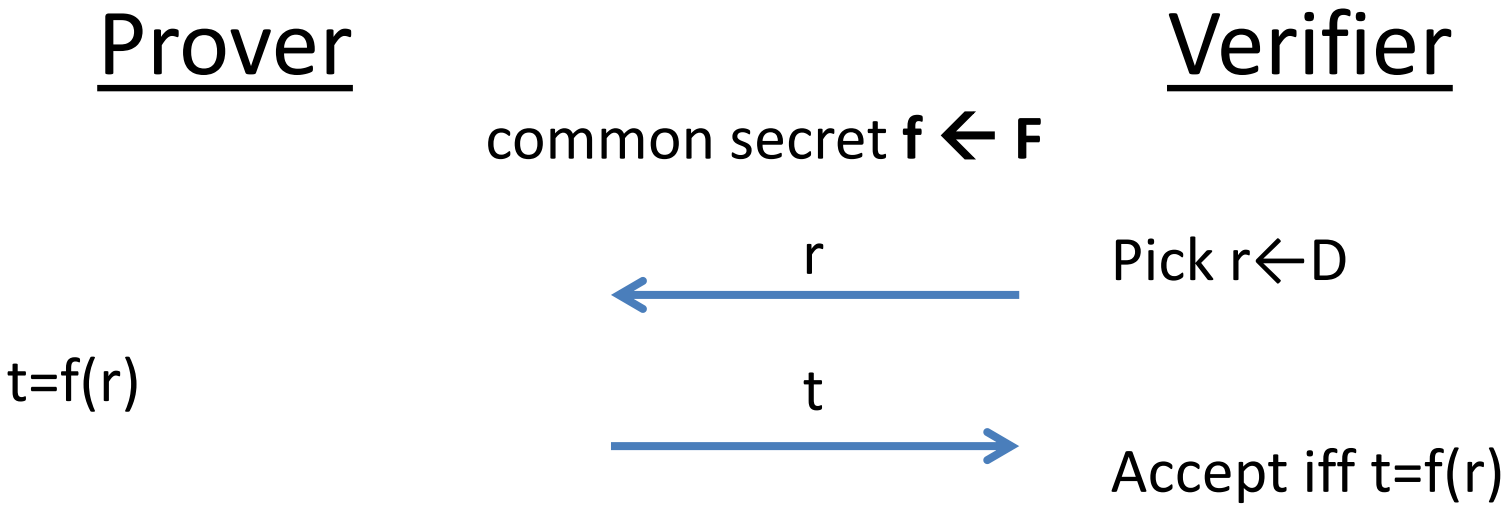
$(d_1, r_1), (d_2, r_2), \ldots$ for r_i randomly chosen from R

Can build a PRF from a weak-PRF using O(n) calls to the weak-PRF [GGM '84, NR '97]

Efficient (1 call to the weak-PRF) actively-secure authentication scheme from any weak-PRF [DKPW '12]

Efficient (1 call to the weak-PRF) MiM secure authentication from any weak-PRF [LM '13]

Passively-Secure Authentication Protocol



Secure against a passive Adversary

(Efficient) Weak-PRF from LPN?

A great open problem! (Can build them based on the related LWE problem [BPR '12])

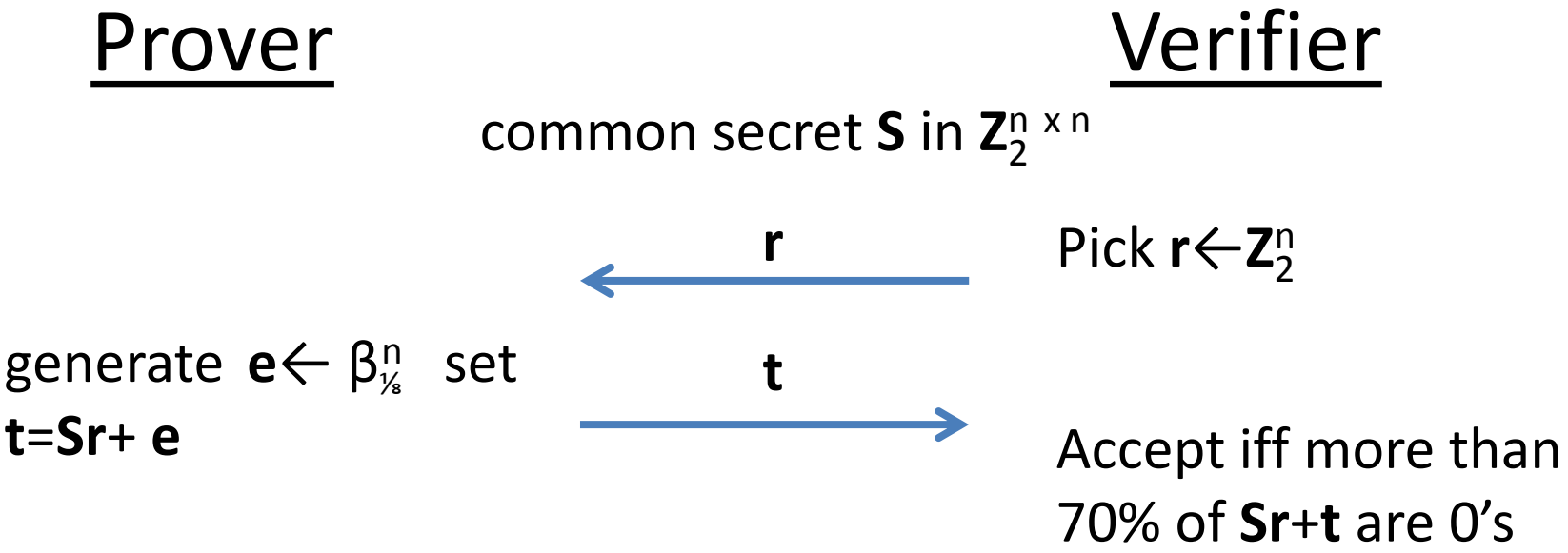
But a "randomized weak-PRF" is possible.

$f_{\mathbf{S}}(\mathbf{d}) = \mathbf{Sd}+\mathbf{e}$

$(\mathbf{d}_1, \mathbf{Sd}_1 + \mathbf{e}_1), (\mathbf{d}_2, \mathbf{Sd}_2 + \mathbf{e}_2), \ldots$ is indistinguishable from uniform

Can build efficient secure authentication from LPN

Passively-Secure Authentication Protocol [HB '01]



As secure as LPN against a passive adversary

"Active" Attack Against HB

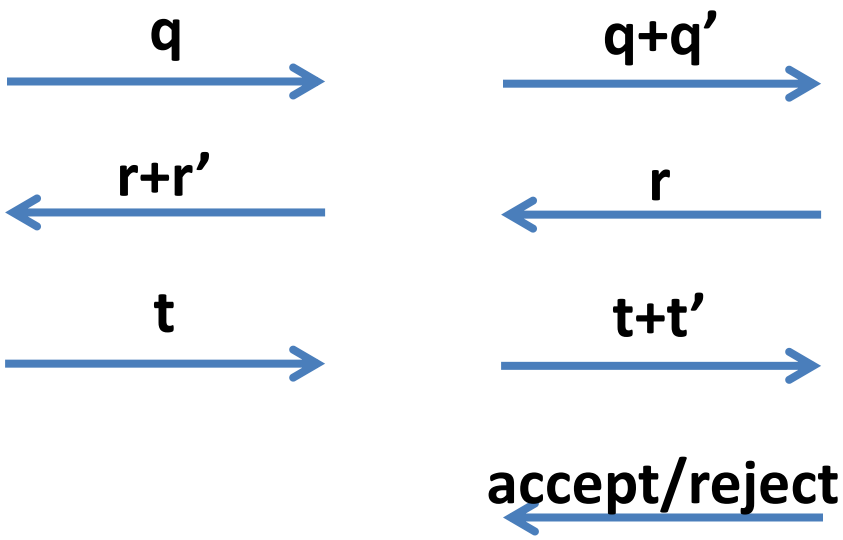
Prover		Verifier
	common secret **S** in $\mathbf{Z}_2^{n \times n}$	
	r = (100 … 0) ←	Pick **r**←$\mathbf{Z}_2^n$
generate **e**← $\beta_{1/8}^n$ set **t**=**Sr**+ **e** = $\mathbf{S_1}$ + **e**	**t** →	Accept iff more than 70% of **Sr**+**t** are 0's

Repeat many times to recover each column of **S** individually

Man-in-the-Middle Security

Prover

Verifier

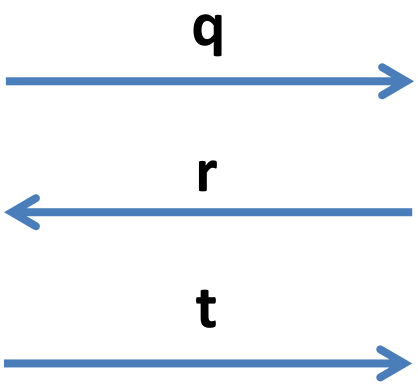


Adversary Phase 1

Man-in-the-Middle Security

Adversary Phase 2

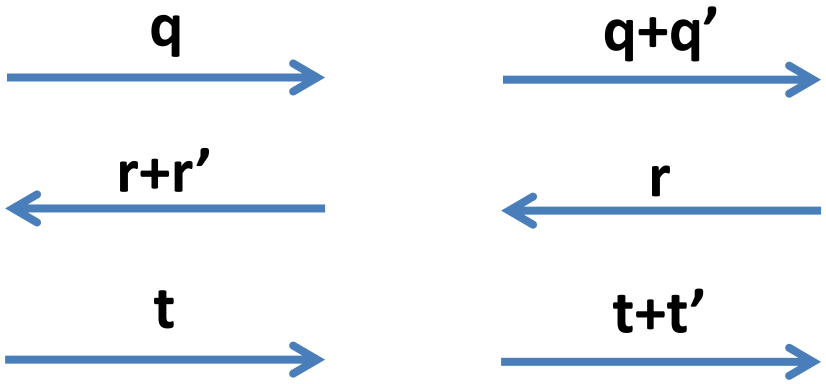
Verifier



A Stronger Requirement

Prover

Verifier



if (**q',r',t'**)≠(**0,0,0**),
Verifier rejects

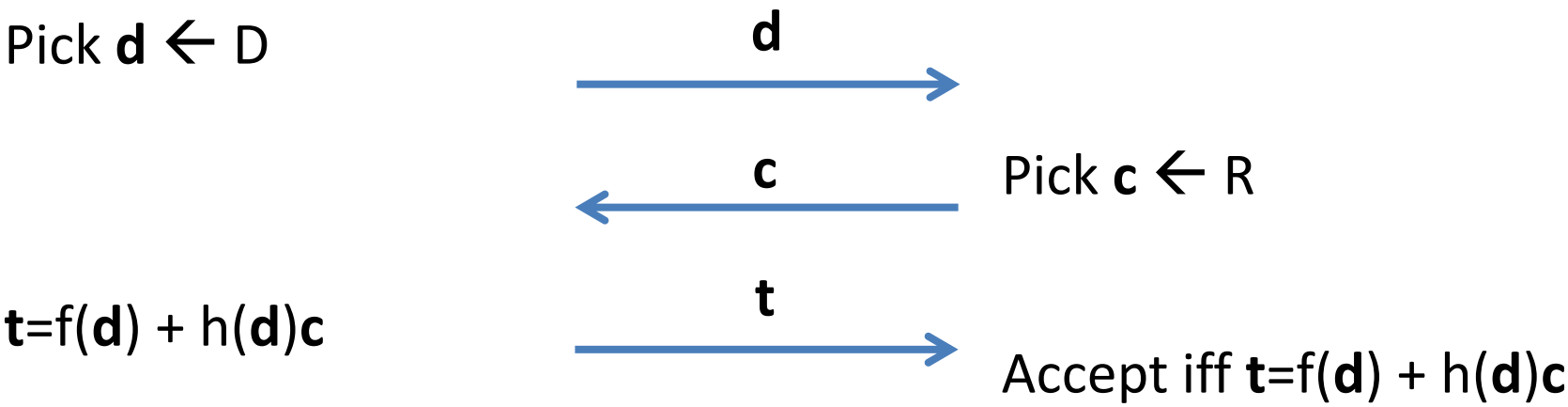
MiM Secure Authentication from any Weak-PRF [LM '13]

A family of weak-PRF functions F: D → R

A family of pairwise-independent functions H: D→ R

Endow R with addition and multiplication to make it a field

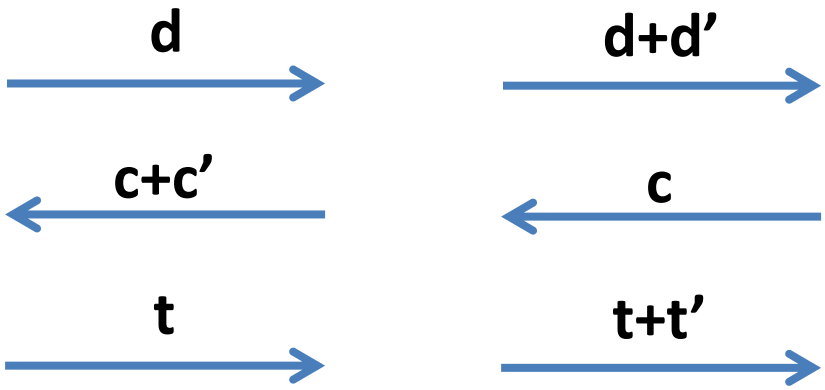
shared key: f ← F, h ← H



Result of the Security Proof

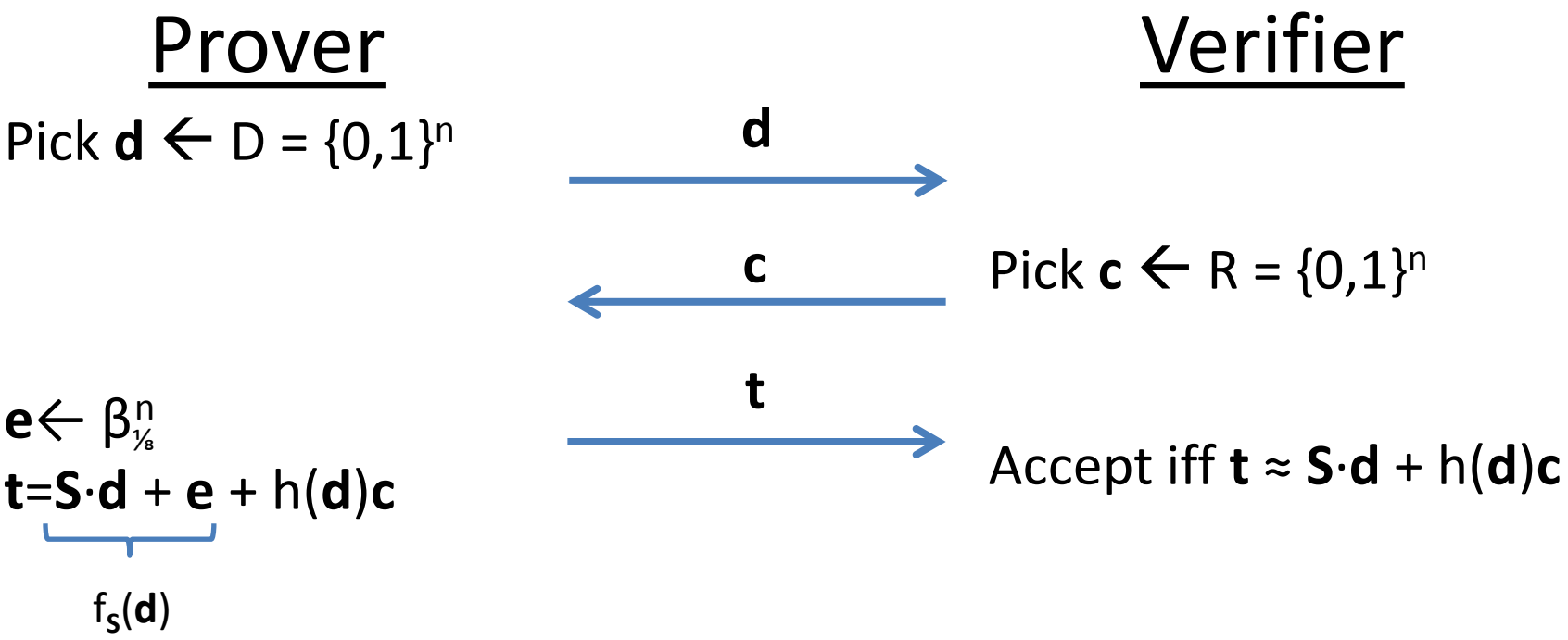
Prover

Verifier



if (**d',r',t'**)≠(**0,0,0**) and
Verifier Accepts,
we can break the weak-PRF

What About LPN?



Easy for the Adversary to modify **t** and have the Verifier accept.

Just add a noise of weight 1 to **t**.

Not an attack, but the proof strategy clearly does not work.

The Fix for Randomized Weak-PRFs [LM '13]

A family of randomized weak-PRF functions F: D → R

A family of pairwise-independent functions H: D→ R

Endow R with addition and multiplication to make it a field

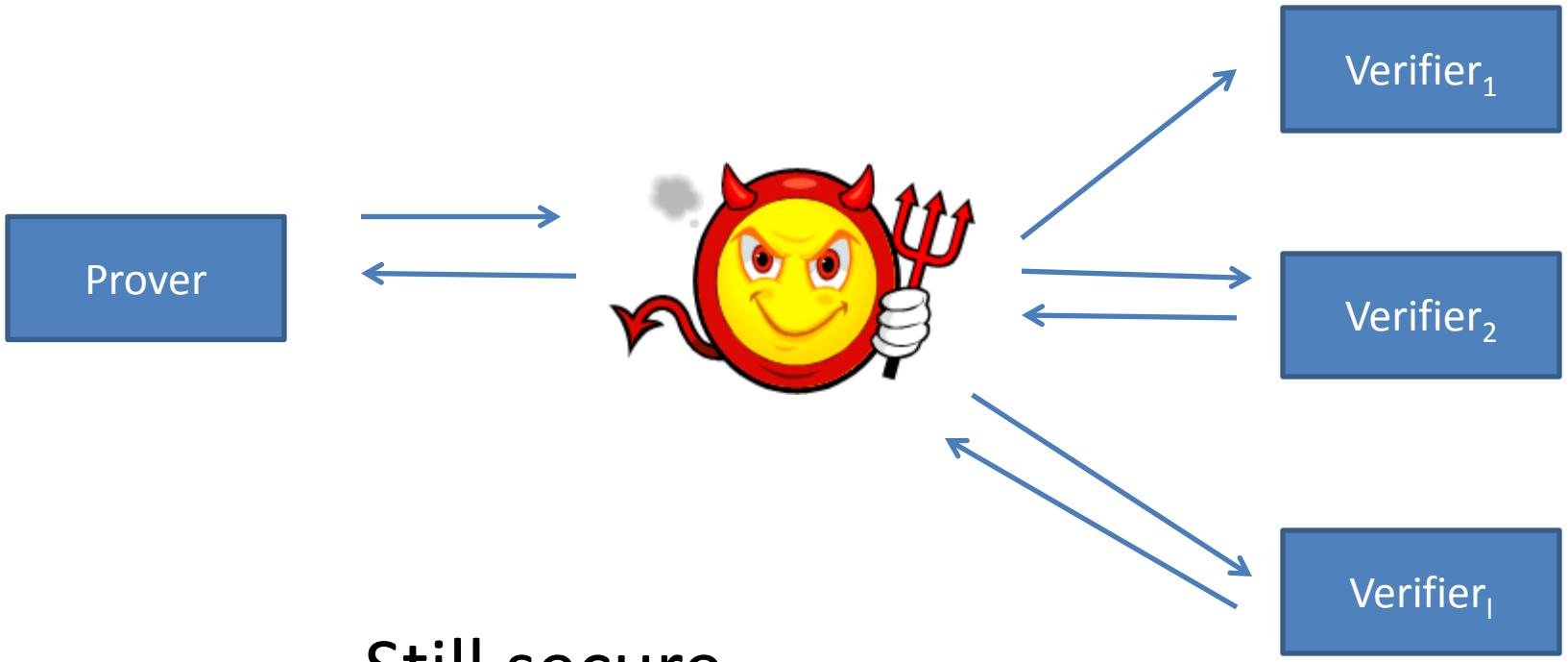
Prover		Verifier
shared key: f ← F, h ← H, **u** ← R		
Pick **d** ← D = $\{0,1\}^n$	**d** →	
	← **c**	Pick **c** ← R = $\{0,1\}^n$
e← $\beta_{1/8}^n$ **t**=(**S**·**d** + **e**)**u** + h(**d**)**c**	**t** →	Accept iff (**t** - h(**d**)**c**)$\mathbf{u}^{-1}$ ≈ **S**·**d**

$f_S(\mathbf{d})$

Even Stronger Security

Security in the concurrent attack model

shared key: $f \leftarrow F$, $h \leftarrow H$

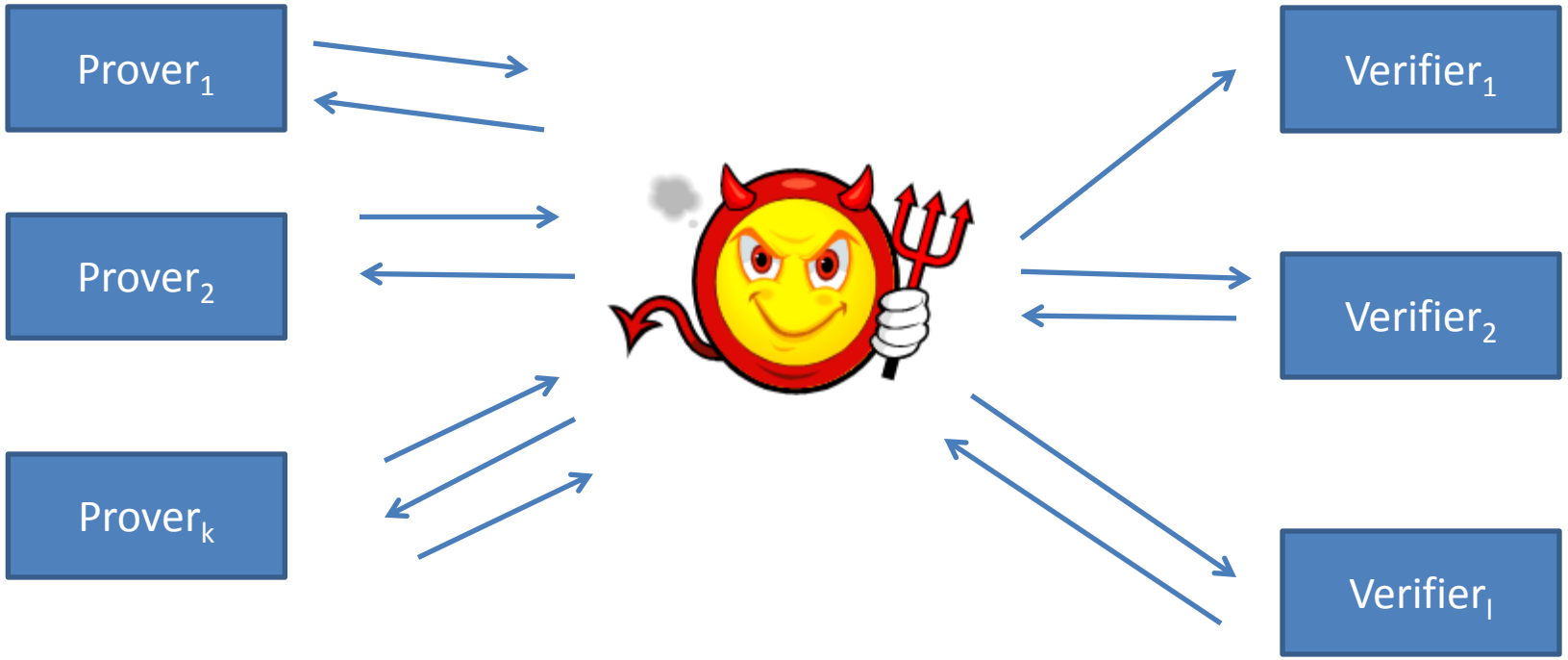


Still secure

Even Stronger Security?

Security in the concurrent attack model.

shared key: $f \leftarrow F$, $h \leftarrow H$



Can get security if H is a (k+1)-wise independent function … but this is not a very satisfactory result. Open problem!

2-Round Authentication / MAC

- Possible from **Key-Homomorphic** Weak-PRFs [KPCJV ’11, DKPW ‘12]

$$a \cdot f_k(x) + b \cdot f_{k'}(x) = f_{ak+bk'}(x)$$

- e.g. $f_k(x) = x^k \bmod p$ is a KHwPRF from DDH
- can build some nice things from them and their randomized versions
- slight modification works for LPN

Efficiency Considerations

We have access to an oracle who has a secret **S** in $\mathbf{Z}_2^{n \times n}$

On every query, the oracle:

1. Picks $\mathbf{r} \leftarrow \mathbf{Z}_2^n$
2. Picks a `noise' $\mathbf{e} \leftarrow \beta_{1/4}^n$ (i.e. e= 0 w.p. ¾ and 1 w.p ¼)
3. Outputs (**r**, **t**=**Sr** + **e**)

S is an n x n matrix – too big!

Idea:

- Make **S** a Toeplitz matrix [GRS '08]

- Open Problems:
  Is decision Toeplitz-LPN as hard as search?
  Can **s** come from the same distribution as **e**?

Ring-LPN [HKLPP '12]

Another idea:

Make the i^{th} column (for i=0 to n-1) of **S** be $\mathbf{s}\mathbf{x}^i$ mod f(x) where f(x) is a degree-n polynomial and **s** is a random polynomial

We have access to an oracle who has a secret **s** in $\mathbf{Z}_2[\mathbf{x}]/(f(x))$
On every query, the oracle:

1. Picks $\mathbf{r} \leftarrow \mathbf{Z}_2[\mathbf{x}]/(f(x))$
2. Picks a `noise' $\mathbf{e} \leftarrow \beta_{1/4}^n$ (i.e. e= 0 w.p. ¾ and 1 w.p ¼)
3. Outputs (**r**, **t**=**sr** + **e**)

- want f(**x**) to be irreducible over Z_2 or split into large factors
- for public-key encryption, want f(**x**) so that |**ab** mod f(**x**)| is not much bigger than |**a**| and |**b**| ($f(\mathbf{x})=\mathbf{x}^{2n}+\mathbf{x}^n+1$ is a good choice for $n=3^k$)
- **s** can have the same distribution as the error

Ring-LPN Open Problems

- Is Ring-LPN hard?
- Are there some irreducible polynomials for which Ring-LPN is easy?
- Is decision Ring-LPN as hard as the search version?

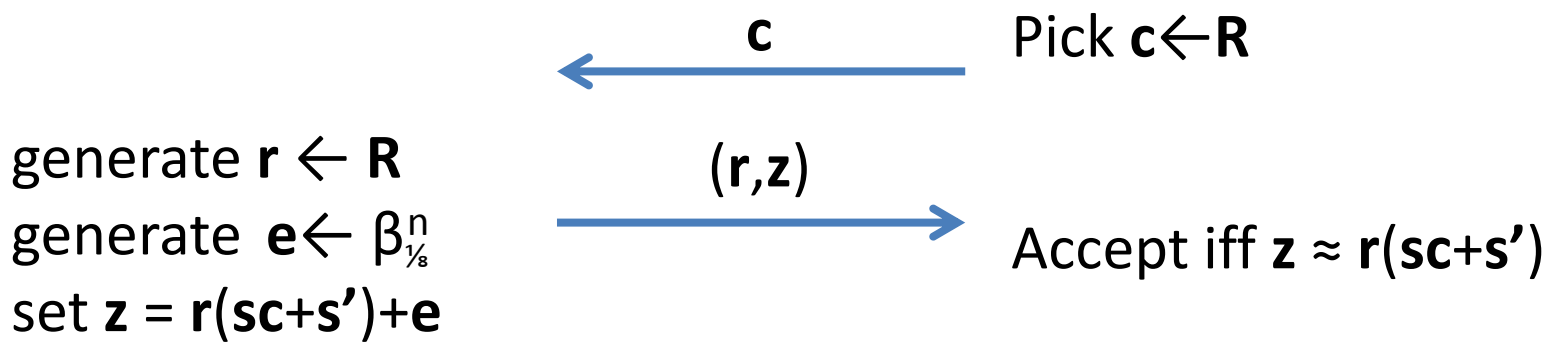
LaPiN [HKLPP '12]

(based on [KPCJV '11])

Prover	Verifier

common secrets **s**, **s'** in $\mathbf{R}=\mathbf{Z}_2[x]/\langle \mathbf{f}(x) \rangle$

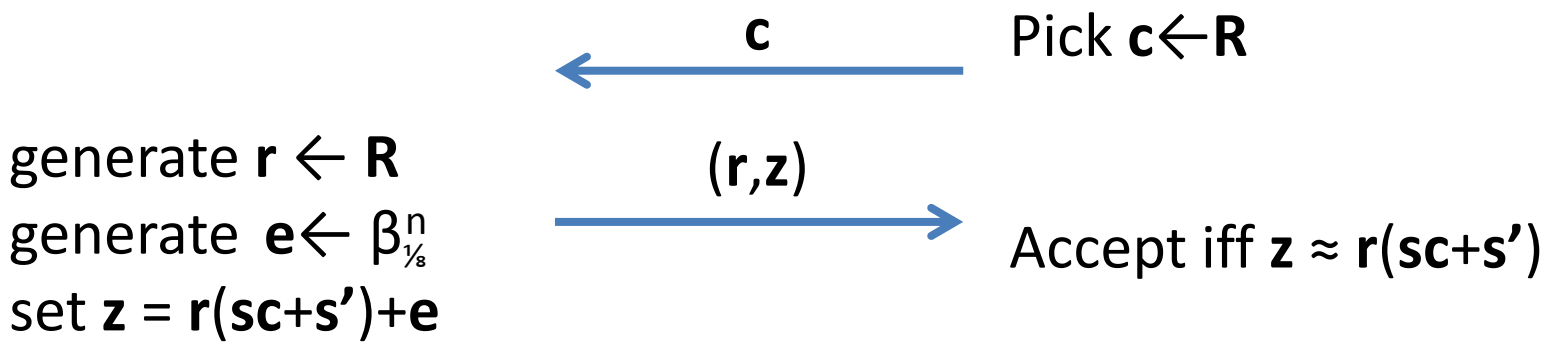
(We will pretend **R** is a field – but we can also work in certain rings)



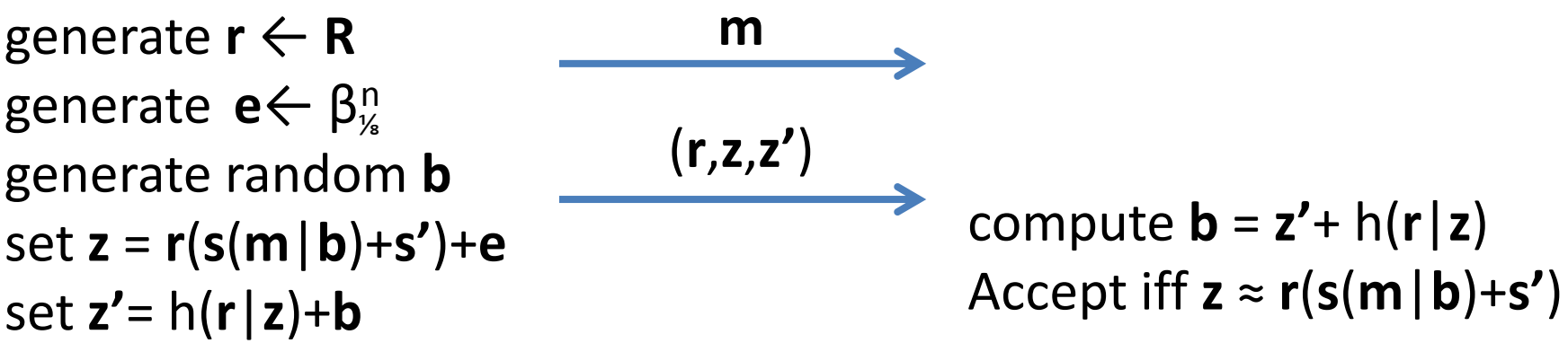
LaPiN Open Problems

- Security against MiM attacks?
  - Open problem
  - Best attack we know runs in time $2^{|C|/2}$ where C is the domain of the challenge **c**
- Interesting direction:
  - Make LaPiN secure against practical side-channel attacks.
  - LaPiN's advantage over AES: it's linear, and so much easier to mask [GLS '14]

Message Authentication [KPCJV ’11, DKPW, 12]



secret keys: **s**,**s’**, pairwise-independent function h



Bibliography

- Avrim Blum, Adam Kalai, Hal Wasserman: Noise-tolerant learning, the parity problem, and the statistical query model. J. ACM 50(4): 506-519 (2003)
- David Wagner: A Generalized Birthday Problem. CRYPTO 2002: 288-303
- Vadim Lyubashevsky: The Parity Problem in the Presence of Noise, Decoding Random Linear Codes, and the Subset Sum Problem. APPROX-RANDOM 2005: 378-389
- Lorenz Minder, Alistair Sinclair: The Extended k-tree Algorithm. J. Cryptology 25(2): 349-382 (2012)

- Avrim Blum, Merrick L. Furst, Michael J. Kearns, Richard J. Lipton: Cryptographic Primitives Based on Hard Learning Problems. CRYPTO 1993: 278-291
- Nicholas J. Hopper, Manuel Blum: Secure Human Identification Protocols. ASIACRYPT 2001: 52-66
- Michael Alekhnovich: More on Average Case vs Approximation Complexity. FOCS 2003: 298-307
- Ari Juels, Stephen A. Weis: Authenticating Pervasive Devices with Human Protocols. CRYPTO 2005: 293-308
- Jonathan Katz, Ji Sun Shin, Adam Smith: Parallel and Concurrent Security of the HB and HB+ Protocols. J. Cryptology 23(3): 402-421 (2010)
- Henri Gilbert, Matthew J. B. Robshaw, Yannick Seurin: HB#: Increasing the Security and Efficiency of HB+. EUROCRYPT 2008: 361-378
- Khaled Ouafi, Raphael Overbeck, Serge Vaudenay: On the Security of HB# against a Man-in-the-Middle Attack. ASIACRYPT 2008: 108-124
- Eike Kiltz, Krzysztof Pietrzak, David Cash, Abhishek Jain, Daniele Venturi: Efficient Authentication from Hard Learning Problems. EUROCRYPT 2011: 7-26
- Yevgeniy Dodis, Eike Kiltz, Krzysztof Pietrzak, Daniel Wichs: Message Authentication, Revisited. EUROCRYPT 2012: 355-374
- Stefan Heyse, Eike Kiltz, Vadim Lyubashevsky, Christof Paar, Krzysztof Pietrzak: Lapin: An Efficient Authentication Protocol Based on Ring-LPN. FSE 2012: 346-365
- Vadim Lyubashevsky, Daniel Masny: Man-in-the-Middle Secure Authentication Schemes from LPN and Weak PRFs. CRYPTO (2) 2013: 308-325
- Lubos Gaspar, Gaetan Leurent, Francois-Xavier Standaert: Hardware Implementation and Side-Channel Analysis of Lapin. CT-RSA 2014