Polly Cracker, Revisited

Martin Albrecht[1] Pooya Farshim[2] Jean-Charles Faugre[1]
Gottfried Herold[3] Ludovic Perret[1]

1 POLSYS Project - INRIA, UPMC, Univ Paris 06
2 Information Security Group, Royal Holloway, University of London
3 Ruhr Universitt Bochum

Bristol, 27.July 2012

Outline

Outline

Homomorphic Encryption

- Homomorphic encryption is a cryptographic primitive which allows to perform arbitrary computation over encrypted data.
- Given a function f and a ciphertext c encrypting a plaintext m, it is possible to transform c to a new ciphertext c' which encrypts $f(m)$.

We can evaluate multivariate (Boolean) polynomials over ciphertexts.

Craig Gentry.
Fully homomorphic encryption using ideal lattices.
In *STOC 09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 169–178, 2009.

An abstract scheme

Let $\mathcal{I} \subset P$ be some ideal in some ring and denote by **Encode**() a function with inverse **Decode**() that maps bit-strings to elements in the quotient ring $P/\mathcal{I}$.

If

$$\textbf{Decode}(\textbf{Encode}(m_0) \ \circ \ \textbf{Encode}(m_1)) = m_0 \ \circ \ m_1 \text{ for } \circ \in \{+, \cdot\},$$

we can encrypt a message m as

$$c = f + \textbf{Encode}(m) \text{ for } f \text{ randomly chosen in } \mathcal{I}.$$

Decryption is equivalent computing remainders modulo $\mathcal{I}$ in P.

Homomorphic features follow from the definition of an ideal.

"Instantiations" of this scheme I

- $P = \mathbb{Z}$
- $\mathcal{I} = \langle p \rangle$ where p is an odd integer
- **Encode**$(\cdot)$ is not injective

Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan.
Fully homomorphic encryption over the integers.
In *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43, 2010.

Approximate GCD

Given $q_i p + r_i$ where $r_i \ll p$, compute p.

"Instantiations" of this scheme II

- $P = \mathbb{F}_q[x_1, \ldots, x_n]$
- $\mathcal{I} = \langle x_1 - s_1, \ldots, x_n - s_n \rangle$
- **Encode**$(\cdot)$ is not injective

Zvika Brakerski and Vinod Vaikuntanathan.
Efficient fully homomorphic encryption from (standard) LWE.
FOCS 2011, 2011.

LWE

Given $\sum_j a_{ij} x_j - \sum_j a_{ij} s_j + e_i$ compute $s_1, \ldots, s_n$.

"Instantiations" of this scheme III

- $P = \mathbb{F}_q[x_1, \ldots, x_n]$
- $\mathcal{I} \approx \langle x_1 - s_1, \ldots, x_n - s_n \rangle$
- **Encode**$(\cdot)$ *is* injective: PoSSo

Neal Koblitz, Alfred J. Menezes, Yi-Hong Wu, and Robert J. Zuccherato.
Algebraic aspects of cryptography.
Springer Verlag, Berlin, Heidelberg, New York, 1998.

PoSSo/GB

Given $\sum h_i f_i$ for $f_i \in \mathcal{I}$ compute the Grbner basis of $\langle f_1, \ldots, f_m \rangle$.

These schemes are known as **Polly Cracker**.

Our contribution

We show that all these schemes can be seen as special instances of problem, which we call **Grbner basis with noise** (GBN).

Put differently, in response to

Boo Barkee, Deh Cac Can, Julia Ecks, Theo Moriarty, and R. F. Ree.

Why you cannot even hope to use Gröbner bases in Public Key Cryptography: An open letter to a scientist who failed and a challenge to those who have not yet failed.
Journal of Symbolic Computations, 18(6):497–501, 1994.

. . . we say: **yes we can!** . . . if we add noise

Outline

Notation & Definitions I

- $P = \mathbb{F}[x_1, \ldots, x_n]$ with some degree-compatible order on monomials.
- $P_{\leq b}$ elements in P of degree at most b.
- $\mathrm{LM}(f)$ is the leading monomial appearing in $f \in P$.
- $\mathrm{LC}(f)$ is the coefficient corresponding to $\mathrm{LM}(f)$ in f.
- $\mathrm{LT}(f)$ is $\mathrm{LC}(f)\mathrm{LM}(f)$.
- d is the degree of Grbner bases in this talk.
- b is the degree of random ideal elements in this talk.

Notation & Definitions II

An example in $\mathbb{F}[x, y, z]$ with term ordering **deglex**:

$$f = 3yz + 2x + 1$$

- $\mathrm{LM}(f) = yz$,
- $\mathrm{LC}(f) = 3$ and
- $\mathrm{LT}(f) = 3yz$.

Notation & Definitions III

Definition (Generated Ideal)

Let $f_1, \ldots, f_m$ be polynomials in P. Define the set

$$\langle f_1, \ldots, f_m \rangle := \left\{ \sum_{i=1}^{m} h_i f_i : h_1, \ldots, h_m \in P \right\}.$$

This set $\mathcal{I}$ is an ideal called the ideal generated by $f_1, \ldots, f_m$.

Notation & Definitions IV

Definition (Gröbner Basis)

Let $\mathcal{I}$ be an ideal of $\mathbb{F}[x_1, \ldots, x_n]$ and fix a monomial ordering. A finite subset

$$G = \{g_1, \ldots, g_m\} \subset \mathcal{I}$$

is said to be a **Gröbner basis** of $\mathcal{I}$ if for any $f \in \mathcal{I}$ there exists $g_i \in G$ with

$$\mathrm{LM}(g_i) \mid \mathrm{LM}(f).$$

- If all f_i linear, then Grbner bases coincide with row echelon forms.
- If all $f_i \in \mathbb{F}[x]$ then Grbner bases coincide with GCDs.

Notation & Definitions V

For each ideal $\mathcal{I}$ and monomial ordering there is a unique **reduced** Grbner basis which can be computed in polynomial time from any Grbner basis.

if you know the Grbner basis you "understand" the ideal.

Grbner bases allow to compute remainders modulo $\mathcal{I}$:

$$f \mod \mathcal{I} = f \mod G.$$

Generating Grbner bases for Crypto I

Definition (S-Polynomial)

The S-polynomial of f and g is defined as

$$S(f,g) = \frac{\mathrm{LCM}(\mathrm{LM}(f),\mathrm{LM}(g))}{\mathrm{LT}(f)} \cdot f - \frac{\mathrm{LCM}(\mathrm{LM}(f),\mathrm{LM}(g))}{\mathrm{LT}(g)} \cdot g.$$

Theorem

A basis $G = \{g_1, \ldots, g_s\}$ for an ideal $\mathcal{I}$ is a Grbner basis if and only if all $S(g_i, g_j)$ reduce to zero by polynomial division.

Generating Grbner bases for Crypto II

begin
- **for** $0 \leq i < n$ **do**
  - **if** $i > n - \ell - 1$ **then**
    - $g_i \leftarrow x_i^d$;
  - **else**
    - $g_i \leftarrow x_i$;
  - **for** $m_j \in M_{<\mathbf{LM}(g_i)}$ **do**
    - $c_{ij} \leftarrow_{\$} \mathbb{F}_q$;
    - $g_i \leftarrow g_i + c_{ij} m_j$;
- **return** $\{g_0, \ldots, g_{n-1}\}$;

Theorem

Let $f, g \in \mathbb{F}[x_0, \ldots, x_{n-1}]$ *with* $a = \mathrm{LM}(f)$ *and* $b = \mathrm{LM}(g)$ *and*

$$\mathrm{LCM}(a, b) = a \cdot b.$$

Then

$$S(f, g) \xrightarrow[\{f,g\}]{} 0.$$

Sampling Elements in $\mathcal{I}$

begin
  $f \leftarrow_{\$} P_{\leq b}$;
  $f \leftarrow f - f \mod G$;
  return f;

Algorithm 1: Sample()

This sampling is uniform for elements $f \in \mathcal{I}$ with $\deg(f) \leq b$ because $P = \mathcal{I} \oplus P/\mathcal{I}$.

Classical Computational Problems I

GB Given access to m samples from $\mathcal{I}$ recover G.

IM Given access to m samples from $\mathcal{I}$ and a challenge $f \in P$, decide if $f \mod G = 0$.

Hardness I

Lemma (IM <=> GB)

If we have an oracle which solves the IM *problem with overwhelming probability, we can construct an algorithm which solves the* GB *problem and vice versa.*

Proof for first direction.

Let $g_i \in G$ arbitrary and let $m_i = \mathrm{LM}(g_i)$

Pick a random $r_i \in P/\mathcal{I}$ and ask for ideal membership of $\tilde{g}_i = m_i + r_i$. If true, then $m_i + r_i \in \langle G \rangle$ with leading monomial m_i. Add it to a list $\tilde{G}$.

Repeat this process for all leading monomials m and all tails r_i.

The list $\tilde{G}$ is a list of elements $\in \langle G \rangle$ with $\mathrm{LM}(\tilde{G}) \supseteq \mathrm{LM}(G)$ which implies $\tilde{G}$ is a Grbner basis.

Hardness II

Assuming that $f_1, \ldots, f_m$ is a random system, the complexity of currently best known algorithms (i.e. with F_5) to solve the GB problem is given by

$$\mathcal{O}\left(\binom{n+D}{D}^{\omega}\right) = \mathcal{O}((n^D)^{\omega})$$

where $2 \leq \omega < 3$ is the linear algebra constant, and D is given by the index of the first non-positive coefficient of:

$$\sum_{k \geq 0} c_k z^k = \frac{(1-z^b)^m}{(1-z)^n}.$$

Thus Grbner bases are exponential in n, if D is polynomial in n.

Hardness III

Definition (GB/IM Assumption)

Let $\mathcal{P}$ be such that $n(\lambda) = \Omega(\lambda)$. Assume $b - d > 0$, $b > 1$, and that $m(\lambda) = c \cdot n(\lambda)$ for a constant $c \geq 1$. Then the advantage of any ppt algorithm in solving the GB/IM problem is negligible as function of λ.

Outline

Symmetric PollyCracker I

$\underline{\mathsf{Gen}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b}(1^\lambda)\textbf{:}}$
begin
  $P \leftarrow_\$ \mathbf{P}_\lambda$;
  $G \leftarrow_\$ \mathsf{GBGen}(1^\lambda, P, d, \ell)$;
  $\mathsf{SK} \leftarrow (G, P, b)$;
  $\mathsf{PK} \leftarrow (P, b)$;
  return $(\mathsf{SK}, \mathsf{PK})$;
end

$\underline{\mathsf{Dec}(\mathsf{c}, \mathsf{SK})}$:
begin
  $\mathsf{m} \leftarrow \mathsf{c} \bmod G$;
  return m;
end

$\underline{\mathsf{Enc}(\mathsf{m}, \mathsf{SK})}$:
begin
  $f \leftarrow_\$ P_{\leq b}$;
  $f' \leftarrow f \bmod G$;
  $f \leftarrow f - f'$;
  $\mathsf{c} \leftarrow \mathsf{m} + f$;
  return c;
end

$\underline{\mathsf{Eval}(\mathsf{c}_0, \ldots, \mathsf{c}_{t-1}, C, \mathsf{PK})}$:
begin
  apply the Add and Mult
    gates of C over P;
  return the result;
end

Figure: The noise-free symmetric Polly Cracker scheme $\mathcal{SPC}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b}$.

Security

The $m(\cdot)$-time IND-CPA security is defined by requiring that the advantage of any ppt $\mathcal{A}$

$$\mathbf{Adv}^{\text{ind-bcpa}}_{m(\cdot),\mathcal{SKE},\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\text{IND-BCPA}^{\mathcal{A}}_{m(\cdot),\mathcal{SKE}}(\lambda) \Rightarrow \mathsf{T}\right] - 1$$

is negligible in λ. The difference with the usual CPA security is that the adversary can query the encryption oracle at most $m(\lambda)$ times.

Theorem

For any $\mathcal{A}$ against the m-time IND-BCPA *security of $\mathcal{SPC}$ there exists a $\mathcal{B}$ against the* IM *problem such that*

$$\mathbf{Adv}^{\text{ind-bcpa}}_{m,\mathcal{SPC},\mathcal{A}}(\lambda) = 2 \cdot \mathbf{Adv}^{\text{im}}_{\mathcal{P},\text{GBGen}(\cdot),d,b,m,\mathcal{B}}(\lambda).$$

Conversely, fory any $\mathcal{A}$ against the IM *problem there exists a $\mathcal{B}$ against the m-time* IND-BCPA *security of $\mathcal{SPC}$ such that*

$$\mathbf{Adv}^{\text{im}}_{\mathcal{P},\text{GBGen}(\cdot),d,b,m,\mathcal{A}}(\lambda) = \mathbf{Adv}^{\text{ind-bcpa}}_{m,\mathcal{SPC},\mathcal{B}}(\lambda).$$

Outline

Conversions in the Literature

- There are a few techniques in the literature, which convert an IND-CPA symmetric additive homomorphic scheme to an IND-CPA public-key additive homomorphic scheme.
- One such conversion is to publish N encryptions of zero $f_1, \ldots, f_N$ and to encrypt as

$$c = \sum_{s \in S} f_s + m$$

where S is a small subset of $\{1, \ldots, N\}$.

While PollyCracker is additive homomorphic and secure up to some bound, none of the proposed conversions give a secure scheme.

Impossibility Result I

Theorem (Dickenstein, Fitchas, Giusti, and Sessa)

Let $\mathcal{I} = \langle f_1, \ldots, f_m \rangle$ *be an ideal in* $P = \mathbb{F}[x_1, \ldots, x_n]$, h *be such that* $\deg(h) \leq D$, *and*

$$h - (h \mod \mathcal{I}) = \sum_{i=1}^{m} h_i f_i,$$

where $h_i \in P$ *and* $\deg(h_i f_i) \leq D$.

Let G *be the output of some Gröbner basis computation algorithm up to degree* D *(i.e., all computations with degree greater than* D *are ignored and dropped).*

Then $h \mod \mathcal{I}$ *can be computed by polynomial reduction of* h *via* G.

Impossibility Result II

Theorem

Let $\mathcal{I} = \langle f_1, \ldots, f_m \rangle$ be an ideal in $P = \mathbb{F}[x_1, \ldots, x_n]$. If there is a ppt algorithm $\mathcal{A}$ which samples elements from $\mathcal{I}$ uniformly given only $(f_1, \ldots, f_m) \in \mathcal{I}$, then there exists a ppt algorithm $\mathcal{B}$ which computes a Grbner basis for $\mathcal{I}$.

Proof.

We can compute the normal forms of any f produced by $\mathcal{A}$ in polynomial time since we know $f_1, \ldots, f_m$.

If f is arbitrary in the ideal $\mathcal{I}$, we know that normals forms are equivalent to Grbner basis computations.

Thus, we have a polynomial time algorithm for computing Grbner bases.

Outline

Foreword

We will now focus on linear Grbner bases for the rest of this talk

- here things thing is nice and easy;
- we can get multiplicative homomorphicity; and
- we have a close relation to LWE.

To see what issues arise for $d > 1$, read [1, 2] or ask during the Q&A.


Gottfried Herold
Polly Cracker, revisited, revisited
PKC 2012, Springer Verlag 2012


with P. Farshim, J.-C. Faugre, G. Herold and L. Perret
Polly Cracker, revisited
full version, in preparation

Discrete Gaussian

Definition (Discrete Gaussian Distribution)

Let $\alpha > 0$ be a real number and $q \in \mathbb{N}$. The discrete Gaussian distribution $\chi_{\alpha,q}$, is a Gaussian distribution rounded to the nearest integer and reduced modulo q with mean zero and standard deviation αq.

Sampling Noisy Elements in $\mathcal{I}$

begin

$f \leftarrow_{\$} P_{\leq b}$;

$f \leftarrow f - (f \mod G)$;

$e \leftarrow_{\$} \chi$;

return $f + e$;

Algorithm 2: Sample()

Noisy Variants of Classical Computational Problems I

GBN Given access to m noisy samples from $\mathcal{I}$ recover G.

IMN Given access to m noisy samples from $\mathcal{I}$ and a challenge $f \in P$, recover if $f \mod G \approx 0$.

Our Ideal Membership with Noise (IMN) is essentially Gentry's Ideal Coset problem for noisy polynomials.

Lemma (IMN Hard $\Leftrightarrow$ GBN Hard)

For any ppt adversary $\mathcal{A}$ against the IMN *problem for $d = 1$ and $q = \mathrm{poly}(n)$, there exists a ppt adversary $\mathcal{B}$ against the* GBN *problem such that*

$$\mathbf{Adv}^{\mathsf{imn}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,\ell,b,\chi,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{\mathsf{gbn}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,\ell,b,\chi,\mathcal{B}}(\lambda).$$

. . . and vice versa.

Proof Sketch.

The proof proceeds as in the GB $\Leftrightarrow$ IM case except that we can amplify our confidence in the output.

Security I

Lemma (LWE Hard $\Rightarrow$ GBN Hard for $d = 1, b = 1$)

Let q be a prime number. Then for any ppt adversary $\mathcal{A}$ against the GBN *problem with $b = d = 1$, there exists a ppt adversary $\mathcal{B}$ against the* LWE *problem such that*

$$\mathbf{Adv}^{\mathsf{gbn}}_{\mathcal{P},\mathsf{GBGen}(\cdot),1,1,\chi,\mathcal{A}}(\lambda) = \mathbf{Adv}^{\mathsf{lwe}}_{n,q,\chi,\mathcal{B}}(\lambda).$$

Proof.

Whenever $\mathcal{A}$ calls its **Sample** oracle, $\mathcal{B}$ queries its own **Sample** oracle to obtain (a, b) where $a = (a_0, \ldots, a_{n-1})$. It returns $\sum a_i x_i - b$ to $\mathcal{A}$. When $\mathcal{A}$ calls its **Finalize** on G, since $d = 1$, we may assume that G is of the form $[x_0 - s_0, \ldots, x_{n-1} - s_{n-1}]$ with $s_i \in \mathbb{F}_q$. Algorithm $\mathcal{B}$ terminates by calling its **Finalize** oracle on $s = (s_0, \ldots, s_{n-1})$.

Security II

Lemma (GBN Hard for $2b \Rightarrow$ GBN Hard for b)

For any ppt adversary $\mathcal{A}$ against the GBN *problem at degree b with noise $\chi_{\alpha,q}$, there exists a ppt adversary $\mathcal{B}$ against the* GBN *problem at degree $2b$ with noise $\chi_{\sqrt{2N}\alpha^2 q,q}$ such that*

$$\mathbf{Adv}^{\mathsf{gbn}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi_{\alpha,q},\mathcal{A}}(\lambda) = \mathbf{Adv}^{\mathsf{gbn}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,2b,\chi_{\sqrt{2N}\alpha^2 q,q},\mathcal{B}}(\lambda)$$

for $N = \binom{n+b}{b}$.

Proof.

Multiply samples f_i, f_j to get $f_{i,j} = f_i \cdot f_j$. To ensure sufficient randomness, sum up $2N$ such products.

Security III

Approximate GCD:

- The GBN problem for $n = 1$ is the approx. GCD problem over $\mathbb{F}_q[x]$.
- This problem has not yet received much attention, and hence it is unclear under which parameters it is hard.
- However, the notion of a Grbner basis can been extended to $\mathbb{Z}[x_0, \ldots, x_{n-1}]$.
- This implies a version of the GBN problem over $\mathbb{Z}$.
- This can be seen as a direct generalisation of the approximate GCD problem in $\mathbb{Z}$.

Security IV

GBN over $\mathbb{F}_2$:

- For $d = 1$ and $q = 2$ we can reduce Max-3SAT instances to GBN instances by translating each clause individually to a Boolean polynomial.
- The Grbner basis returned by an arbitrary algorithm $\mathcal{A}$ solving GBN using a **bounded number** of samples will provide a solution to the Max-3SAT problem.
- Vice versa, we may convert a GBN problem for $d = 1$ to a Max-SAT problem (more precisely Partial Max-Sat) by running an ANF to CNF conversion algorithm.

Security V

Best known attack (for $d = 1$):

- We reduce GBN to a larger LWE instance.
- Denote by $N = \binom{n+b}{b}$ the number of monomials up to degree b.
- Let $\mathcal{M} : P \to \mathbb{F}_q^N$ be a function which maps polynomials in P to vectors in $\mathbb{F}_q^N$ by assigning the i-th component of the image vector the coefficient of the i-th monomial $\in M_{\leq b}$.
- Reply to each **Sample** query by the LWE oracle by calling the GBN **Sample** oracle to retrieve f, compute $v = \mathcal{M}(f)$ and return (a, b) with $a = (v_{N-1}, \ldots, v_1)$ and $b = -v_0$.
- When the LWE oracle queries its **Finalize** with s query the GBN **Finalize** with $[x_0 - s_0, \ldots, x_{n-1} - s_{n-1}]$.

Outline

Polly Cracker with Noise

- GBN/IMN allow to construct a noisy version of our symmetric Polly Cracker scheme: $\mathcal{SPCN}$.
- $\mathcal{SPCN}$ is IND-CPA under the GBN assumption.
- Using any symmetric-to-asymmetric conversion from literature this leads to a public-key Polly Cracker scheme.
- This scheme is somewhat homomorphic and can support a fixed but arbitrary number of multiplications.
- This also implies that Regev's public-key scheme based on LWE is multiplicative homomorphic under some choice of parameters.

$$\begin{aligned} c_0 \cdot c_1 &= (m_0 + 2e_1 + \sum h_{0i} g_i) \cdot (m_1 + 2e_1 + \sum h_{1i} g_i) \\ &= m_0 m_1 + 2e_0 e_1 + 2\tilde{e} + \sum \tilde{h}_i g_i \end{aligned}$$

Outline

Abstract Nonsense?

Generalising and unifying known things is fun, but is it useful? That is, do results from one instantiation carry over to another?

Arora&Ge's algorithm for LWE

Compute $h_i = f_i \cdot \prod_{j=1}^{|r|} ((f_i + j) \cdot (f_i - j))$ for samples f_i.

Compute the Grbner basis of $\langle h_0, \ldots, h_{m-1} \rangle$.

Chen&Nguyen's algorithm for AGCD

Compute $h_1 = f \cdot \prod_{j=1}^{|r|} ((f + j) \cdot (f - j)) \bmod h_0$ for noisy sample f and clean sample h_0.

Compute the Grbner basis of $\langle h_0, h_1 \rangle$.

Ring-LWE I

Compute $a_i^{-1} \cdot (a_i \cdot s + e_i) \approx s$ where all computations are $\bmod x^n + 1$.

> *Given that this new approach allows one to cast both LWE and approximate GCD in the same framework, can one also capture ring-LWE.*
>
> – Bristol Cryptography Blog

To compute a_i^{-1} in $P = \mathbb{F}_q[x]/\langle x^n + 1\rangle$ we run the extended GCD algorithm which returns (g, v, w) for inputs a, b such that

$$g = v \cdot a + w \cdot a.$$

Hence, for our inputs it will compute

$$1 = v \cdot a_i + w \cdot (x^n + 1) \text{ and thus } v \equiv a_i^{-1} \mod x^n + 1.$$

Ring-LWE II

In the language of Grbner bases the extended GCD equivalent is often called "lifting".

Given an ideal $I = (f_1, ..., f_r)$ and some $g \in \mathcal{I}$, find $s_1, \ldots, s_r$ such that $g = s_1 f_1 + \cdots + s_r f_r$.

- The problem is easy given a Grbner basis $g_1, \ldots, g_r$ (in our case $x^n + 1$), since every element $h \in \langle g_1, \ldots, g_r \rangle$ can be written as $h = \sum h_i \cdot g_i$ where $\mathrm{LM}(h_i g_i) \leq \mathrm{LM}(h)$.
- In general, it is hard because the degree of the output may be large.

In any case, instead of solving solving GBN, we are now lifting with GBN, i.e., we keep track of our computation.

Ring-LWE III

Example:

```
sage: P.<x,y,z> = PolynomialRing(GF(127),order='deglex')
sage: I = Ideal(P.random_element() for _ in range(4))
sage: s = I.gens()
sage: I.groebner_basis()
[x - 24, y - 20, z - 17]
sage: b = P.random_element()
sage: b -= b.reduce(I)
sage: b in I
True
sage: a = b.lift(s)
sage: sum(a[i]*s[i] for i in range(s))
16*x*z + 50*y^2 + 53*z^2 - 57*x + 36
sage: b
16*x*z + 50*y^2 + 53*z^2 - 57*x + 36
```

Given tuples (a, b) find s.

Thank you for your attention

Questions?