PollyCracker Revisited

Martin Albrecht[1] Pooya Farshim[2] Jean-Charles Faugère[1]
Ludovic Perret[1]

1 SALSA Project - INRIA, UPMC, Univ Paris 06
2 Information Security Group, Royal Holloway, University of London

25. May 2011

Outline

Outline

Homomorphic Encryption I

- From an algebraic perspective, homomorphic encryption can be seen as the ability to evaluate multivariate (Boolean) polynomials over ciphertexts.
- Hence, an instantiation of homomorphic encryption over the ring of multivariate polynomials itself is perhaps the most natural.

Homomorphic Encryption II

- Let $\mathcal{I} \subset P = \mathbb{F}[x_0, \ldots, x_{n-1}]$ be some ideal and denote by **Encode**() an injective function, with inverse **Decode**(), that maps bits to elements in the quotient ring $P/\mathcal{I}$.
- Assume that **Decode**(**Encode**(m_0) $\circ$ **Encode**(m_1)) $= m_0 \circ m_1$ for $\circ \in \{+, \cdot\}$.
- We can encrypt a message m as

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

- Decryption is performed by computing remainders modulo $\mathcal{I}$.

Homomorphic Encryption III

- This construction is somewhat homomorphic

$$\begin{aligned} c_0 + c_1 &= f_0 + \mathbf{Encode}(m_0) + f_1 + \mathbf{Encode}(m_1) \\ &= f + \mathbf{Encode}(m_0) + \mathbf{Encode}(m_1) \text{ for some } f \in \mathcal{I}. \end{aligned}$$

$$\begin{aligned} c_0 \cdot c_1 &= (f_0 + \mathbf{Encode}(m_0)) \cdot (f_1 + \mathbf{Encode}(m_1)) \\ &= f_0 \cdot f_1 + f_0 \cdot \mathbf{Encode}(m_1) + f_1 \cdot \mathbf{Encode}(m_0) \\ &\quad + \mathbf{Encode}(m_0) \cdot \mathbf{Encode}(m_1) \\ &= f + \mathbf{Encode}(m_0) \cdot \mathbf{Encode}(m_1) \text{ for some } f \in \mathcal{I}. \end{aligned}$$

- This construction is **Polly Cracker**.

Homomorphic Encryption IV

- However, our confidence in Polly Cracker-style schemes has been shaken as almost all such proposals are broken.
- It is a long standing open research challenge to propose a secure Polly Cracker-style encryption scheme,
- . . . even better if we can make it somewhat homomorphic.

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.

Outline

Notation I

- $P = \mathbb{F}[x_0, \ldots, x_{n-1}]$ with some ordering 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 $\mathrm{LC}(f)\mathrm{LM}(f)$.

Notation 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 III

Definition (Generated Ideal)

Let $f_0, \ldots, f_{m-1}$ be polynomials in P. Define the set

$$\langle f_0, \ldots, f_{m-1} \rangle := \left\{ \sum_{i=0}^{m-1} h_i f_i : h_0, \ldots, h_{m-1} \in P \right\}.$$

This set $\mathcal{I}$ is an ideal called the ideal generated by $f_0, \ldots, f_{m-1}$.

Notation IV

Definition (Gröbner Basis)

Let $\mathcal{I}$ be an ideal of $\mathbb{F}[x_0, \ldots, x_{n-1}]$ and fix a monomial ordering. A finite subset

$$G = \{g_0, \ldots, g_{m-1}\} \subset \mathcal{I}$$

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

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

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

Gröbner bases allow to compute remainders modulo $\mathcal{I}$: $r = f \mod \mathcal{I} = f \mod G$.

Characterisation of Gröbner bases I

Definition (S-Polynomial)

The S-polynomial of f and g is defined as

$$S(f,g) \;=\; \frac{x^\gamma}{\mathrm{LT}(f)} \cdot f \;-\; \frac{x^\gamma}{\mathrm{LT}(g)} \cdot g.$$

where

$$x^\gamma \;=\; \mathrm{LCM}(\mathrm{LM}(f), \mathrm{LM}(g)).$$

Characterisation of Gröbner bases II

Definition (t-Representation)

Fix a monomial order and let $F = \{f_0, \ldots, f_{m-1}\} \subset P$ be an **unordered** set of polynomials and let t be a monomial. Given a polynomial $f \in P$, we say that f has a t-**representation** if f can be written in the form

$$f = h_0 f_0 + \cdots + h_{m-1} f_{m-1},$$

such that whenever $h_i f_i \neq 0$, we have $h_i f_i \leq t$.

Furthermore, we write that $f \xrightarrow[F]{} 0$ if and only if f has an $\mathrm{LM}(f)$-representation with respect to F.

Characterisation of Gröbner bases III

Theorem

A basis $G = \{g_0, \ldots, g_{s-1}\}$ *for an ideal* I *is a Gröbner basis if and only if*

$$S(g_i, g_j) \xrightarrow[G]{} 0$$

for all $i \neq j$.

Outline

Generating Gröbner bases

1 **begin**
2 **for** $0 \leq i < n$ **do**
3 **for** $0 \leq j < M_{<x_i^d}$ **do**
4 $c_{ij} \leftarrow_{\$} \mathbb{F}_q$;
5 $g_i \leftarrow x_i^d + \sum_j c_{ij} m_j$;
6 **return** $(g_0, \ldots, g_{n-1})$;
7 **end**

Algorithm 1: $\mathsf{GBGen}_{\mathsf{dense}}(1^\lambda, P, d)$

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

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

Formalising the Problems I

proc. Initialize$(1^\lambda, \mathcal{P}, d)$:
begin
  $P \leftarrow_\$ \mathbf{P}_\lambda$;
  $G \leftarrow_\$ \mathrm{GBGen}(1^\lambda, P, d)$;
  return $(1^\lambda, P)$;
end

proc. Sample():
begin
  $f \leftarrow_\$ P_{\leq b}$;
  $f \leftarrow f - (f \mod G)$;
  return f;
end

proc. Finalize(G'):
begin
  return $(G = G')$;
end

Figure: Game $\mathrm{GB}_{\mathcal{P},\mathrm{GBGen}(\cdot),d,b,m}$. An adversary is valid if it calls the **Sample** oracle at most $m(\lambda)$ times.

Formalising the Problems II

Definition (Gröbner Basis (GB) Problem)

The advantage of a ppt algorithm $\mathcal{A}$ in solving the Gröbner basis problem with respect to basis generation algorithm $\mathsf{GBGen}(\cdot)$ is defined by

$$\mathbf{Adv}^{\mathsf{gb}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m(\cdot),\mathcal{A}}(\lambda) := \Pr\left[\mathsf{GB}^{\mathcal{A}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m(\cdot)}(\lambda) \Rightarrow \mathsf{T}\right],$$

where game $\mathsf{GB}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m(\cdot)}$ is shown in Figure 1.

Formalising the Problems III

proc. Initialize$(1^\lambda, \mathcal{P}, d)$:
begin
  $P \leftarrow_\$ \mathbf{P}_\lambda$;
  $G \leftarrow_\$ \mathsf{GBGen}(1^\lambda, P, d)$;
  return $(1^\lambda, P)$;
end

proc. Sample():
begin
  $f \leftarrow_\$ P_{\leq b}$;
  $f \leftarrow f - (f \mod G)$;
  return f;
end

proc. Challenge():
begin
  $f \leftarrow_\$ P_{\leq b}$;
  return f;
end

proc. Finalize(r'):
begin
  return $(r' = f \mod G)$;
end

Figure: Game $\mathrm{IR}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m}$. An adversary is valid if it calls the **Sample** oracle at most $m(\lambda)$ times.

Formalising the Problems IV

Definition (Ideal Remainder (IR) Problem)

The advantage of a ppt algorithm $\mathcal{A}$ in solving the ideal remainder problem is defined by

$$\mathbf{Adv}^{\mathsf{ir}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m(\cdot),\mathcal{A}}(\lambda) := \Pr\left[\mathsf{IR}^{\mathcal{A}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m(\cdot)}(\lambda) \Rightarrow \mathsf{T}\right] - 1/C(\lambda),$$

where game $\mathsf{IR}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m(\cdot)}$ is shown in Figure 2.

Formalising the Problems V

proc. Initialize$(1^\lambda, \mathcal{P}, d)$:

begin
- $P \leftarrow_\$ \mathbf{P}_\lambda$;
- $G \leftarrow_\$ \mathrm{GBGen}(1^\lambda, P, d)$;
- $c \leftarrow_\$ \{0,1\}$;
- **return** $(1^\lambda, P)$;

end

proc. Sample():

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

end

proc. Challenge():

begin
- $f \leftarrow_\$ P_{\leq b}$;
- **if** $c = 1$ **then**
  - $f \leftarrow f - (f \mod G)$;
- **return** f;

end

proc. Finalize(c'):

begin
- **return** $(c = c')$;

end

Figure: Game $\mathrm{IM}_{\mathcal{P},\mathrm{GBGen}(\cdot),d,b,m}$. An adversary is valid if it calls the **Sample** oracle at most $m(\lambda)$ times.

Formalising the Problems VI

Definition (Ideal Membership (IM) Problem)

The advantage of a ppt algorithm $\mathcal{A}$ in solving the ideal membership problem is defined by

$$\mathbf{Adv}^{\mathsf{im}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m(\cdot),\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\mathsf{IM}^{\mathcal{A}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m(\cdot)}(\lambda) \Rightarrow \mathsf{T}\right] - 1,$$

where game $\mathsf{IM}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m(\cdot)}$ is shown in Figure 3.

Note

We can view the IM problem as the decisional version of the IR problem.

Hardness I

Lemma (IR <=> GB)

For any ppt adversary $\mathcal{A}$ against the IR *problem, there exists a ppt adversary $\mathcal{B}$ against the* GB *problem such that*

$$\mathbf{Adv}^{\mathsf{ir}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m,\mathcal{A}}(\lambda)^{\mathrm{poly}(\lambda)} \leq \mathbf{Adv}^{\mathsf{gb}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m,\mathcal{B}}(\lambda).$$

Conversely, for any ppt adversary $\mathcal{B}$ against the GB *problem, there exists a ppt adversary $\mathcal{A}$ against the* IR *problem such that*

$$\mathbf{Adv}^{\mathsf{gb}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m,\mathcal{B}}(\lambda) = \mathbf{Adv}^{\mathsf{ir}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,m,\mathcal{A}}(\lambda).$$

Hardness II

Proof for first direction.

Consider an arbitrary element g_i in the Gröbner basis G. We can write g_i as $m_i + \tilde{g}_i$ for some $\tilde{g}_i < g_i$ and $m_i = \mathrm{LM}(g_i)$.

Now, assume the normal form of m_i is r_i and suppose that $r_i < m_i$. This implies that $m_i = \sum_{j=0}^{n-1} h_j g_j + r_i$ for some $h_i \in P$. Hence, we have $m_i - r_i \in \langle G \rangle$: an element $\in \langle G \rangle$ with leading monomial m_i.

Repeat this process for all monomials up to and including degree d and accumulate the results $m_i - r_i$ in a list $\tilde{G}$.

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 Gröbner basis.

We cannot amplify our confidence since we only have a limited number of samples.

Hardness III

IR <=> IM

When the search space of remainders is $\mathrm{poly}(\lambda)$, the IM and IR problems are equivalent, since the attacker can exhaustively search for the remainder using the IM oracle.

Thus, we have decision to search reduction for some parameters.

Hardness IV

Assuming that $f_0, \ldots, f_{m-1}$ 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 Gröbner bases are exponential in n, if D is polynomial in n.

Hardness V

Corollary

Let $c \geq 0$. Then for $m(\lambda) = c \cdot n(\lambda)$ or $m(\lambda) = c \cdot n(\lambda)^b$ polynomials of degree b in some ideal $\mathcal{I}$, the Gröbner basis of $\mathcal{I}$ can be computed in exponential or polynomial time in $n(\lambda)$ respectively.

Definition (GB/IR/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/IR/IM problem is negligible as function of λ.

Outline

Symmetric PollyCracker I

Algo. $\mathsf{Gen}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b}(1^\lambda)$

begin
- $P \leftarrow_\$ \mathbf{P}_\lambda$;
- $G \leftarrow_\$ \mathsf{GBGen}(1^\lambda, P, d)$;
- $\mathsf{SK} \leftarrow (G, P, b)$;
- $\mathsf{PK} \leftarrow (P, b)$;
- **return** $(\mathsf{SK}, \mathsf{PK})$;

end

Algo. $\mathsf{Enc}(m, \mathsf{SK})$:

begin
- $f \leftarrow_\$ P_{\leq b}$;
- $\leftarrow f - (f \mod G)$;
- $c \leftarrow m + f$;
- **return** c;

end

Algo. $\mathsf{Dec}(c, \mathsf{SK})$:

begin
- $m \leftarrow c \mod G$;
- **return** m;

end

Algo. $\mathsf{Eval}(c_0, \ldots, c_{t-1}, C, \mathsf{PK})$:

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 I

The $m(\cdot)$-time IND-CPA security of a (homomorphic) symmetric-key encryption scheme is defined in the usual way by requiring that the advantage of any probabilistic polynomial-time adversary $\mathcal{A}$

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

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

Security II

Theorem

Let $\mathcal{A}$ be a ppt adversary against the m-time IND-BCPA *security of the scheme described in Figure 4. Then there exists a ppt adversary $\mathcal{B}$ against the* IM *problem such that for all $\lambda \in \mathbb{N}$ we have*

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

Conversely, let $\mathcal{A}$ be a ppt adversary against the IM *problem. Then there exists a ppt adversary $\mathcal{B}$ against the m-time* IND-BCPA *security of the scheme described in Figure 4 such that for all $\lambda \in \mathbb{N}$ we have*

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

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 $f_0, \ldots, f_{N-1}$ and to encrypt as

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

where S is a subset of $\{0, \ldots, N-1\}$.

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 $\mathcal{I} = \langle f_0, \ldots, f_{m-1} \rangle$ *be an ideal in* $P = \mathbb{F}[x_0, \ldots, x_{n-1}]$, h *be such that* $\deg(h) \leq D$, *and*

$$h - (h \mod \mathcal{I}) = \sum_{i=0}^{m-1} h_i f_i,$$

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 $\mathcal{I} = \langle f_0, \ldots, f_{m-1} \rangle$ be an ideal in $P = \mathbb{F}[x_0, \ldots, x_{n-1}]$. If there is a ppt algorithm $\mathcal{A}$ which samples elements from $\mathcal{I}$ uniformly given only $(f_0, \ldots, f_{m-1}) \in \mathcal{I}$, then there exists a ppt algorithm $\mathcal{B}$ which computes a Gröbner basis for $\mathcal{I}$.

Proof.

We can compute the normal forms of any f produced by $\mathcal{A}$ in polynomial time since we know $f_0, \ldots, f_{m-1}$. If f is arbitrary in the ideal $\mathcal{I}$, we know that normals forms are equivalent to Gröbner basis computations. Thus, we have a polynomial time algorithm for computing Gröbner bases.

Outline

Discrete Gaussian

A noise distribution χ will parametrise various games below. The discrete Gaussian distribution is of particular interest to us.

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.

Gröbner Bases with Noise I

proc. Initialize$(1^\lambda, \mathcal{P}, d)$:

begin

$P \leftarrow_\$ \mathbf{P}_\lambda$;

$G \leftarrow_\$ \mathsf{GBGen}(1^\lambda, P, d)$;

return $(1^\lambda, P)$;

end

proc. Sample():

begin

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

$e \leftarrow_\$ \chi$;

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

return f;

end

proc. Finalize(G'):

begin

$\tilde{G} \leftarrow$ reduced GB of G;

$\tilde{G}' \leftarrow$ reduced GB of G';

return $\tilde{G} = \tilde{G}'$;

end

Figure: Game $\mathsf{GBN}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi}$.

Gröbner Bases with Noise II

Definition (Gröbner Basis with Noise (GBN) Problem)

The Gröbner Basis with Noise Problem is defined through game $\mathrm{GBN}_{\mathcal{P},\mathrm{GBGen}(\cdot),d,b,\chi}$ as shown in Figure 5. The advantage of a ppt algorithm $\mathcal{A}$ in solving the GBN problem is

$$\mathbf{Adv}^{\mathrm{gbn}}_{\mathcal{P},\mathrm{GBGen}(\cdot),d,b,\chi,\mathcal{A}}(\lambda) := \Pr\left[\mathrm{GBN}^{\mathcal{A}}_{\mathcal{P},\mathrm{GBGen}(\cdot),d,b,\chi}(\lambda) \Rightarrow \mathsf{T}\right].$$

Note that we do not impose a restriction on the number of samples any more.

Ideal Remainders with Noise I

proc. Initialize$(1^\lambda, \mathcal{P}, d)$:

begin
- $P \leftarrow_\$ \mathbf{P}_\lambda$;
- $G \leftarrow_\$ \mathsf{GBGen}(1^\lambda, P, d)$;
- **return** $(1^\lambda, P)$;

end

proc. Sample():

begin
- $f \leftarrow_\$ P_{\leq b}$;
- $e \leftarrow_\$ \chi$;
- $f \leftarrow f - (f \mod G) + e$;
- **return** f;

end

proc. Challenge():

begin
- $f \leftarrow_\$ P_{\leq b}$;
- **return** f;

end

proc. Finalize(r'):

begin
- **return** $(r' = f \mod G)$;

end

Figure: Game $\mathsf{IRN}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi}$.

Ideal Remainders with Noise II

Definition (Ideal Remainder with Noise (IRN) Problem)

The Ideal Remainder with Noise Problem is defined through game $\mathsf{IRN}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi}$ as shown in Figure 6. The advantage of a ppt algorithm $\mathcal{A}$ in solving the IRN problem is

$$\mathbf{Adv}^{\mathsf{irn}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi,\mathcal{A}}(\lambda) := \Pr\left[\mathsf{IRN}^{\mathcal{A}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi}(\lambda) \Rightarrow \mathsf{T}\right] - 1/C(\lambda).$$

Lemma (IRN Hard $\Leftrightarrow$ GBN Hard)

For any ppt adversary $\mathcal{A}$ against the IRN *problem, there exists a ppt adversary $\mathcal{B}$ against the* GBN *problem such that*

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

. . . and vice versa.

Ideal Membership with Noise (Ideal Coset) I

proc. Initialize$(1^\lambda, \mathcal{P}, d)$:

begin
- $P \leftarrow_\$ \mathbf{P}_\lambda$;
- $G \leftarrow_\$ \mathsf{GBGen}(1^\lambda, P, d)$;
- $c \leftarrow_\$ \{0,1\}$;
- **return** $(1^\lambda, P)$;

end

proc. Sample():

begin
- $f \leftarrow_\$ P_{\leq b}$;
- $e \leftarrow_\$ \chi$;
- $f \leftarrow f - (f \mod G) + e$;
- **return** f;

end

proc. Challenge():

begin
- $f, e \leftarrow_\$ P_{\leq b}, \chi$;
- **if** $c = 0$ **then**
  - $f \leftarrow f - (f \mod G) + e$;
- **return** f;

end

proc. Finalize(c'):

begin
- **return** $(c' = c)$;

end

Figure: Game $\mathsf{IMN}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi}$.

Ideal Membership with Noise (Ideal Coset) II

Definition (Ideal Membership with Noise (IMN) Problem)

The Ideal Membership with Noise (IMN) Problem is defined as a game, denoted $\mathsf{IMN}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi}$, shown in Figure 7. The advantage of a ppt algorithm $\mathcal{A}$ in solving the ideal membership with noise problem is defined by

$$\mathbf{Adv}^{\mathsf{imn}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi,\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\mathsf{IMN}^{\mathcal{A}}_{\mathcal{P},\mathsf{GBGen}(\cdot),d,b,\chi}(\lambda) \Rightarrow \mathsf{T}\right] - 1.$$

Lemma (IMN Hard $\Leftrightarrow$ IRN Hard)

For any ppt adversary $\mathcal{A}$ against the IMN *problem, there exists a ppt adversary $\mathcal{B}$ against the* IRN *problem such that*

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

if $q(\lambda)^{\dim_{\mathbb{F}_q}(\mathcal{P}(\lambda)/\mathsf{GBGen}(\cdot))}$ *is polynomial in* λ.

. . . and vice versa.

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 $\chi_{\sqrt{N}\alpha^2 q,q}$ such that*

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

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 N 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 Gröbner 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 Gröbner 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 $[x_0 - s_0, \ldots, x_{n-1} - s_{n-1}]$.

Polly Cracker with Noise

- GBN/IRN/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.

Remark

We implemented a toy version of this scheme.

Thank you for your attention

Questions?