On Practical and Asymptotic Complexity of Solving Generic Systems of Equations

Lesser-Known Factoids from a Decade of Work

Bo-Yin Yang

Institute of Information Science
Academia Sinica, Taipei, Taiwan
by@crypto.tw

March 2, 2013

Co-workers

- Chen-Mou Cheng
- Tung Chou
- Ruben Niederhagen
- ... many others over the years

Difficulty of System-Solving?

- C Shannon (1947): "Cracking a good cryptosystem should be as difficult as solving a large non-linear system of equations."

Difficulty of System-Solving?

- C Shannon (1947): "Cracking a good cryptosystem should be as difficult as solving a large non-linear system of equations."
- Solving Multivariate Quadratic systems ($\mathcal{MQ}$) is NP-Hard.

Difficulty of System-Solving?

- C Shannon (1947): "Cracking a good cryptosystem should be as difficult as solving a large non-linear system of equations."
- Solving Multivariate Quadratic systems ($\mathcal{MQ}$) is NP-Hard.

$\mathcal{MQ}(m, n, q)$ Problem

Find a solution to a system of m equations in n variables over $\mathbb{F}_q$.

Difficulty of System-Solving?

- C Shannon (1947): "Cracking a good cryptosystem should be as difficult as solving a large non-linear system of equations."
- Solving Multivariate Quadratic systems ($\mathcal{MQ}$) is NP-Hard.

$\mathcal{MQ}(m, n, q)$ Problem

Find a solution to a system of m equations in n variables over $\mathbb{F}_q$.

clearly $\mathcal{MQ}$ can be extended to higher degrees.

Difficulty of System-Solving?

- C Shannon (1947): "Cracking a good cryptosystem should be as difficult as solving a large non-linear system of equations."
- Solving Multivariate Quadratic systems ($\mathcal{MQ}$) is NP-Hard.

$\mathcal{MQ}(m, n, q)$ Problem

Find a solution to a system of m equations in n variables over $\mathbb{F}_q$.

clearly $\mathcal{MQ}$ can be extended to higher degrees.

- Solving Non-Linear Multivariate systems is conjectured to be exponentially hard in probability.

Difficulty of System-Solving?

- C Shannon (1947): "Cracking a good cryptosystem should be as difficult as solving a large non-linear system of equations."
- Solving Multivariate Quadratic systems ($\mathcal{MQ}$) is NP-Hard.

$\mathcal{MQ}(m, n, q)$ Problem

Find a solution to a system of m equations in n variables over $\mathbb{F}_q$.

clearly $\mathcal{MQ}$ can be extended to higher degrees.

- Solving Non-Linear Multivariate systems is conjectured to be exponentially hard in probability. To be precise,

Hardness Assumption of QUAD (Patarin et al)

For any (probabilistic) Turing machine $\mathcal{A}$ trying to solve an MQ system with randomly drawn coefficients where $m/n = c + o(1)$, and sub-exponential function $\eta(n)$, the probability that $\mathcal{A}$ returns the correct answer in time $\eta(n)$ is negligible.

The point of this talk

Conventional Wisdom among cryptographers

$\mathbf{F_4}/\mathbf{F_5}$ is the best general-purpose solver of multivariate nonlinear systems, complexity of system-solving should be estimated using $\mathbf{F_4}/\mathbf{F_5}$.

The point of this talk

Conventional Wisdom among cryptographers

$\mathbf{F_4}/\mathbf{F_5}$ is the best general-purpose solver of multivariate nonlinear systems, complexity of system-solving should be estimated using $\mathbf{F_4}/\mathbf{F_5}$.

Discordant Notes

- Some claim generally superior results, e.g.
  - MutantXL (Ding et al),
  - G2V/GVW (Gao et al).
- Some claim good results for special algorithms (e.g., SAT solvers) or special situations (sparse or clumped equations).

The point of this talk

Conventional Wisdom among cryptographers

$\mathbf{F_4}/\mathbf{F_5}$ is the best general-purpose solver of multivariate nonlinear systems, complexity of system-solving should be estimated using $\mathbf{F_4}/\mathbf{F_5}$.

Discordant Notes

- Some claim generally superior results, e.g.
  - MutantXL (Ding et al),
  - G2V/GVW (Gao et al).
- Some claim good results for special algorithms (e.g., SAT solvers) or special situations (sparse or clumped equations).

Random Systems, where $\frac{m}{n} = c + o(1)$

- For $\mathbb{F}_2$, Brute-force often practically best, otherwise, asymptotically, XL with sparse solver is best.

The point of this talk

Conventional Wisdom among cryptographers

$\mathbf{F_4}/\mathbf{F_5}$ is the best general-purpose solver of multivariate nonlinear systems, complexity of system-solving should be estimated using $\mathbf{F_4}/\mathbf{F_5}$.

Discordant Notes

- Some claim generally superior results, e.g.
  - MutantXL (Ding et al),
  - G2V/GVW (Gao et al).
- Some claim good results for special algorithms (e.g., SAT solvers) or special situations (sparse or clumped equations).

Random Systems, where $\frac{m}{n} = c + o(1)$

- For $\mathbb{F}_2$, Brute-force often practically best, otherwise, asymptotically, XL with sparse solver is best.
- For large fields and $c > 1$, XL with Sparse Solver often best.

Introduction to XL

- First suggested by D. Lazard: Gröbner bases, gaussian elimination and resolution of systems of algebraic equations, *EUROCAL '83*
- Rediscovered by N. Courtois, A. Klimov, J. Patarin, and A. Shamir: Efficient algorithms for solving overdefined systems of multivariate polynomial equations, *Eurocrypt 2000*
- **eXtends** a polynomial system by multiplying appropriate monomials and **Linearizes** by treating each monomial as an independent variable
  - Turns system solving into linear algebra

Basic XL at Degree D

Let $\mathcal{T}^{(D)} := \{\deg \leq D \text{ monomials}\}$, $T := |\mathcal{T}^{(D)}|$.

- eXtend: first multiply each l_i of degree d_i by every monomial $\mathbf{x}^{\mathbf{b}} := x_1^{b_1} \cdots x_n^{b_n} \in \mathcal{T}^{(D-d_i)}$ to get equations $\mathcal{R}^{(D)}$.
- Linearize: then solve $\mathcal{R}^{(D)}$ as a linear system in all the $\mathbf{x}^{\mathbf{b}} \in \mathcal{T}^{(D)}$. We may be able to solve the system or to reduce down to a univariate equation (say in x_1).

Basic XL at Degree D

Let $\mathcal{T}^{(D)} := \{\deg \leq D \text{ monomials}\}$, $T := |\mathcal{T}^{(D)}|$.

- eXtend: first multiply each l_i of degree d_i by every monomial $\mathbf{x}^{\mathbf{b}} := x_1^{b_1} \cdots x_n^{b_n} \in \mathcal{T}^{(D-d_i)}$ to get equations $\mathcal{R}^{(D)}$.
- Linearize: then solve $\mathcal{R}^{(D)}$ as a linear system in all the $\mathbf{x}^{\mathbf{b}} \in \mathcal{T}^{(D)}$. We may be able to solve the system or to reduce down to a univariate equation (say in x_1).

Criterion for Success

- If the linearized system leads to $1 = 0$ then system is inconsistent.

Basic XL at Degree D

Let $\mathcal{T}^{(D)} := \{\text{deg} \leq D \text{ monomials}\}$, $T := |\mathcal{T}^{(D)}|$.

- eXtend: first multiply each l_i of degree d_i by every monomial $\mathbf{x^b} := x_1^{b_1} \cdots x_n^{b_n} \in \mathcal{T}^{(D-d_i)}$ to get equations $\mathcal{R}^{(D)}$.
- Linearize: then solve $\mathcal{R}^{(D)}$ as a linear system in all the $\mathbf{x^b} \in \mathcal{T}^{(D)}$. We may be able to solve the system or to reduce down to a univariate equation (say in x_1).

Criterion for Success

- If the linearized system leads to $1 = 0$ then system is inconsistent.
- In the notation of Courtois et al, $R := |\mathcal{R}^{(D)}|$ and I counts resp. eqs and indep eqs among $\mathcal{R}^{(D)}$.

Basic XL at Degree D

Let $\mathcal{T}^{(D)} := \{\text{deg} \leq D \text{ monomials}\}$, $T := |\mathcal{T}^{(D)}|$.

- eXtend: first multiply each l_i of degree d_i by every monomial $\mathbf{x^b} := x_1^{b_1} \cdots x_n^{b_n} \in \mathcal{T}^{(D-d_i)}$ to get equations $\mathcal{R}^{(D)}$.
- Linearize: then solve $\mathcal{R}^{(D)}$ as a linear system in all the $\mathbf{x^b} \in \mathcal{T}^{(D)}$. We may be able to solve the system or to reduce down to a univariate equation (say in x_1).

Criterion for Success

- If the linearized system leads to $1 = 0$ then system is inconsistent.
- In the notation of Courtois et al, $R := |\mathcal{R}^{(D)}|$ and I counts resp. eqs and indep eqs among $\mathcal{R}^{(D)}$. If $T = I$ then system is inconsistent.

Basic XL at Degree D

Let $\mathcal{T}^{(D)} := \{\deg \leq D \text{ monomials}\}$, $T := |\mathcal{T}^{(D)}|$.

- eXtend: first multiply each l_i of degree d_i by every monomial $\mathbf{x^b} := x_1^{b_1} \cdots x_n^{b_n} \in \mathcal{T}^{(D-d_i)}$ to get equations $\mathcal{R}^{(D)}$.
- Linearize: then solve $\mathcal{R}^{(D)}$ as a linear system in all the $\mathbf{x^b} \in \mathcal{T}^{(D)}$. We may be able to solve the system or to reduce down to a univariate equation (say in x_1).

Criterion for Success

- If the linearized system leads to $1 = 0$ then system is inconsistent.
- In the notation of Courtois et al, $R := |\mathcal{R}^{(D)}|$ and I counts resp. eqs and indep eqs among $\mathcal{R}^{(D)}$. If $T = I$ then system is inconsistent.
- If the linear system stays consistent, then we have a solution. If for some D, $T - I = 1$, then we have a unique solution.

Basic XL at Degree D

Let $\mathcal{T}^{(D)} := \{\deg \leq D \text{ monomials}\}$, $T := |\mathcal{T}^{(D)}|$.

- eXtend: first multiply each f_i of degree d_i by every monomial $\mathbf{x}^{\mathbf{b}} := x_1^{b_1} \cdots x_n^{b_n} \in \mathcal{T}^{(D-d_i)}$ to get equations $\mathcal{R}^{(D)}$.
- Linearize: then solve $\mathcal{R}^{(D)}$ as a linear system in all the $\mathbf{x}^{\mathbf{b}} \in \mathcal{T}^{(D)}$. We may be able to solve the system or to reduce down to a univariate equation (say in x_1).

Criterion for Success

- If the linearized system leads to $1 = 0$ then system is inconsistent.
- In the notation of Courtois et al, $R := |\mathcal{R}^{(D)}|$ and I counts resp. eqs and indep eqs among $\mathcal{R}^{(D)}$. If $T = I$ then system is inconsistent.
- If the linear system stays consistent, then we have a solution. If for some D, $T - I = 1$, then we have a unique solution.
- Courtois et al notes that if $1 < T - I < \min(D, q-1)$ then we can reduce to univariate equations.

Basic XL at Degree D

Let $\mathcal{T}^{(D)} := \{\text{deg} \leq D \text{ monomials}\}$, $T := |\mathcal{T}^{(D)}|$.

- eXtend: first multiply each l_i of degree d_i by every monomial $\mathbf{x}^{\mathbf{b}} := x_1^{b_1} \cdots x_n^{b_n} \in \mathcal{T}^{(D-d_i)}$ to get equations $\mathcal{R}^{(D)}$.
- Linearize: then solve $\mathcal{R}^{(D)}$ as a linear system in all the $\mathbf{x}^{\mathbf{b}} \in \mathcal{T}^{(D)}$. We may be able to solve the system or ~~to reduce down to a univariate equation (say in x_1).~~

Criterion for Success

- If the linearized system leads to $1 = 0$ then system is inconsistent.
- In the notation of Courtois et al, $R := |\mathcal{R}^{(D)}|$ and I counts resp. eqs and indep eqs among $\mathcal{R}^{(D)}$. If $T = I$ then system is inconsistent.
- If the linear system stays consistent, then we have a solution. If for some D, $T - I = 1$, then we have a unique solution.
- ~~Courtois et al notes that if $1 < T - I < \min(D, q-1)$ then we can reduce to univariate equations.~~ **Just doesn't happen!**

Basic XL at Degree D

Let $\mathcal{T}^{(D)} := \{\deg \leq D \text{ monomials}\}$, $T := |\mathcal{T}^{(D)}|$.

- eXtend: first multiply each l_i of degree d_i by every monomial $\mathbf{x}^{\mathbf{b}} := x_1^{b_1} \cdots x_n^{b_n} \in \mathcal{T}^{(D-d_i)}$ to get equations $\mathcal{R}^{(D)}$.
- Linearize: then solve $\mathcal{R}^{(D)}$ as a linear system in all the $\mathbf{x}^{\mathbf{b}} \in \mathcal{T}^{(D)}$. We may be able to solve the system or express all the highest degree monomials $\mathcal{T}^{(=D)}$ in $\mathcal{T}^{(D-1)}$.

Criterion for Success

- If the linearized system leads to $1 = 0$ then system is inconsistent.
- In the notation of Courtois et al, $R := |\mathcal{R}^{(D)}|$ and I counts resp. eqs and indep eqs among $\mathcal{R}^{(D)}$. If $T = I$ then system is inconsistent.
- If the linear system stays consistent, then we have a solution. If for some D, $T - I = 1$, then we have a unique solution.

Can use sparse solvers (e.g. Wiedemann) if ≤ 1 solutions expected.

Basic XL at Degree D

Let $\mathcal{T}^{(D)} := \{\deg \leq D \text{ monomials}\}$, $T := |\mathcal{T}^{(D)}|$.

- eXtend: first multiply each l_i of degree d_i by every monomial $\mathbf{x}^{\mathbf{b}} := x_1^{b_1} \cdots x_n^{b_n} \in \mathcal{T}^{(D-d_i)}$ to get equations $\mathcal{R}^{(D)}$.
- Linearize: then solve $\mathcal{R}^{(D)}$ as a linear system in all the $\mathbf{x}^{\mathbf{b}} \in \mathcal{T}^{(D)}$. We may be able to solve the system or express all the highest degree monomials $\mathcal{T}^{(=D)}$ in $\mathcal{T}^{(D-1)}$.

Criterion for Success

- If the linearized system leads to $1 = 0$ then system is inconsistent.
- In the notation of Courtois et al, $R := |\mathcal{R}^{(D)}|$ and I counts resp. eqs and indep eqs among $\mathcal{R}^{(D)}$. If $T = I$ then system is inconsistent.
- If the linear system stays consistent, then we have a solution. If for some D, $T - I = 1$, then we have a unique solution.

Can use sparse solvers (e.g. Wiedemann) if ≤ 1 solutions expected. Can even throw away random rows to get square matrix probabilistically.

More Advanced Variations of XL

XL: boost up to degree D *then run an elimination on the resulting extended Macaulay Matrix*.

More Advanced Variations of XL

XL: boost up to degree D *then run an elimination on the resulting extended Macaulay Matrix*. Can continue onto higher degrees by multiplying result with variables.

More Advanced Variations of XL

XL: boost up to degree D *then run an elimination on the resulting extended Macaulay Matrix*. Can continue onto higher degrees by multiplying result with variables.

XL2: Suppose we only manage to eliminate the top level monomials, then multiply repeatedly by other variables to raise effective degrees.

More Advanced Variations of XL

XL: boost up to degree D *then run an elimination on the resulting extended Macaulay Matrix*. Can continue onto higher degrees by multiplying result with variables.

XL2: Suppose we only manage to eliminate the top level monomials, then multiply repeatedly by other variables to raise effective degrees. Maybe the same as "MutantXL".

More Advanced Variations of XL

XL: boost up to degree D *then run an elimination on the resulting extended Macaulay Matrix*. Can continue onto higher degrees by multiplying result with variables.

XL2: Suppose we only manage to eliminate the top level monomials, then multiply repeatedly by other variables to raise effective degrees. (Rediscovery Courtois-Patarin 2002)

$\mathbf{F_4}$: Improvement of XL2 using some selection criteria.

More Advanced Variations of XL

XL: boost up to degree D *then run an elimination on the resulting extended Macaulay Matrix*. Can continue onto higher degrees by multiplying result with variables.

XL2: Suppose we only manage to eliminate the top level monomials, then multiply repeatedly by other variables to raise effective degrees. (Rediscovery Courtois-Patarin 2002)

$\mathbf{F_4}$: Improvement of XL2 using some selection criteria.

$\mathbf{F_5}$: Even more improvement over $\mathbf{F_4}$, can potentially avoid any excess polynomials being generated.

More Advanced Variations of XL

XL: boost up to degree D *then run an elimination on the resulting extended Macaulay Matrix*. Can continue onto higher degrees by multiplying result with variables.

XL2: Suppose we only manage to eliminate the top level monomials, then multiply repeatedly by other variables to raise effective degrees. (Rediscovery Courtois-Patarin 2002)

$\mathbf{F_4}$: Improvement of XL2 using some selection criteria.

$\mathbf{F_5}$: Even more improvement over $\mathbf{F_4}$, can potentially avoid any excess polynomials being generated.

Functional Equivalence of XL2/$\mathbf{F_4}$/$\mathbf{F_5}$

They all work at the same degree; also possible to track the corresponding degree in XL.

More Advanced Variations of XL

XL: boost up to degree D *then run an elimination on the resulting extended Macaulay Matrix*. Can continue onto higher degrees by multiplying result with variables.

XL2: Suppose we only manage to eliminate the top level monomials, then multiply repeatedly by other variables to raise effective degrees. (Rediscovery Courtois-Patarin 2002)

$\mathbf{F_4}$: Improvement of XL2 using some selection criteria.

$\mathbf{F_5}$: Even more improvement over $\mathbf{F_4}$, can potentially avoid any excess polynomials being generated.

Functional Equivalence of XL2/$\mathbf{F_4}$/$\mathbf{F_5}$

They all work at the same degree; also possible to track the corresponding degree in XL. Note: XL2 (MutantXL) is primitive form of $\mathbf{F_4}/\mathbf{F_5}$ $\Leftrightarrow$ $\mathbf{F_4}/\mathbf{F_5}$ is more advanced form of XL2 (hence XL).

Facts of Life for XL

$$\#\text{ monomials: } T = [t^D]\left((1-t^q)^n(1-t)^{-(n+1)}\right); \tag{1}$$

$$\#\text{ free monoms: } T-I \geq [t^D]\left(\frac{(1-t^q)^n}{(1-t)^{n+1}}\prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right). \tag{2}$$

Here $\deg l_i := d_i$, $[u]s :=$ coefficient of u in expansion of s.

Facts of Life for XL

$$\#\text{ monomials: } T = [t^D]\left((1-t^q)^n(1-t)^{-(n+1)}\right); \tag{1}$$

$$\#\text{ free monoms: } T-I \geq [t^D]\left(\frac{(1-t^q)^n}{(1-t)^{n+1}}\prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right). \tag{2}$$

Here $\deg l_i := d_i$, $[u]s :=$ coefficient of u in expansion of s. Assuming some usually-true regularity conditions on the (l_i) Eq. 2 is $=$ as long as RHS remains positive;

Facts of Life for XL

$$\#\text{ monomials: } T = [t^D]\left((1-t^q)^n(1-t)^{-(n+1)}\right); \tag{1}$$

$$\#\text{ free monoms: } T-I \geq [t^D]\left(\frac{(1-t^q)^n}{(1-t)^{n+1}}\prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right). \tag{2}$$

Here $\deg l_i := d_i$, $[u]s :=$ coefficient of u in expansion of s. Assuming some usually-true regularity conditions on the (l_i) Eq. 2 is $=$ as long as RHS remains positive; solution expected at $D_{XL} = \min\{D : (\text{RHS of Eq. 2}) \leq 0\}$.

Facts of Life for XL

$$\#\text{ monomials: } T = [t^D]\left((1-t^q)^n(1-t)^{-(n+1)}\right); \tag{1}$$

$$\#\text{ free monoms: } T-I \geq [t^D]\left(\frac{(1-t^q)^n}{(1-t)^{n+1}}\prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right). \tag{2}$$

Here $\deg l_i := d_i$, $[u]s :=$ coefficient of u in expansion of s. Assuming some usually-true regularity conditions on the (l_i) Eq. 2 is $=$ as long as RHS remains positive; solution expected at $D_{XL} = \min\{D : (\text{RHS of Eq. 2}) \leq 0\}$.

$$T = \binom{n+D}{D}, \quad T-I = [t^D]\left((1-t)^{m-n-1}(1+t)^m\right)$$

is the reduced case for large fields ($q > D$).

Facts of Life for XL

$$\#\text{ monomials: } T = [t^D]\left((1-t^q)^n(1-t)^{-(n+1)}\right); \tag{1}$$

$$\#\text{ free monoms: } T-I \geq [t^D]\left(\frac{(1-t^q)^n}{(1-t)^{n+1}}\prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right). \tag{2}$$

Here $\deg l_i := d_i$, $[u]s :=$ coefficient of u in expansion of s. Assuming some usually-true regularity conditions on the (l_i) Eq. 2 is $=$ as long as RHS remains positive; solution expected at $D_{XL} = \min\{D : (\text{RHS of Eq. 2}) \leq 0\}$.

$$T = \binom{n+D}{D}, \quad T-I = [t^D]\left((1-t)^{m-n-1}(1+t)^m\right)$$

is the reduced case for large fields ($q > D$).
$C_{XL} \approx 3kT^2(c_0 + c_1 \lg T)$ using modified Wiedemann algorithms (k is average number of terms per equation).

Facts of Life for XL

$$\#\text{ monomials: } T = [t^D]\left((1-t^q)^n(1-t)^{-(n+1)}\right); \tag{1}$$

$$\#\text{ free monoms: } T-I \geq [t^D]\left(\frac{(1-t^q)^n}{(1-t)^{n+1}}\prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right). \tag{2}$$

Here $\deg l_i := d_i$, $[u]s :=$ coefficient of u in expansion of s. Assuming some usually-true regularity conditions on the (l_i) Eq. 2 is $=$ as long as RHS remains positive; solution expected at $D_{XL} = \min\{D : (\text{RHS of Eq. 2}) \leq 0\}$.

$$T = \binom{n+D}{D}, \quad T-I = [t^D]\left((1-t)^{m-n-1}(1+t)^m\right)$$

is the reduced case for large fields ($q > D$).
$C_{XL} \approx 3kT^2(c_0 + c_1 \lg T)$ using modified Wiedemann algorithms (k is average number of terms per equation). T is singly exponential if $D \propto n$.

$\mathbf{F_4}/\mathbf{F_5}$/XL2 vs XL for generic systems

Small Fields ($T = [t^D]\ \left((1-t^q)^n(1-t)^{-(n+1)}\right)$;)

$$D_{XL} = \min\left\{D : [t^D]\left(\frac{(1-t^q)^n}{(1-t)^{n+1}}\prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right) \leq 0\right\}.$$

$$D_{reg} = \min\left\{D : [t^D]\left(\frac{(1-t^q)^n}{(1-t)^{n}}\prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right) < 0\right\}.$$

$\mathbf{F_4}/\mathbf{F_5}$/XL2 vs XL for generic systems

Small Fields ($T = [t^D]\ \left((1-t^q)^n(1-t)^{-(n+1)}\right)$;)

$$D_{XL} = \min\left\{D : [t^D]\ \left(\frac{(1-t^q)^n}{(1-t)^{n+1}}\ \prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right) \le 0\right\}.$$

$$D_{reg} = \min\left\{D : [t^D]\ \left(\frac{(1-t^q)^n}{(1-t)^{n}}\ \prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right) < 0\right\}.$$

Large Fields ($q > D$, $T = \binom{n+D}{D}$)

$$D_{XL} = \min\left\{D : [t^D]\ \left((1-t)^{m-n-1}\,(1+t)^m\right) \le 0\right\},$$

$$D_{reg} = \min\left\{D : [t^D]\ \left((1-t)^{m-n}\,(1+t)^m\right) < 0\right\}.$$

$\mathbf{F_4}/\mathbf{F_5}$/XL2 vs XL for generic systems

Small Fields ($T = [t^D]\ \left((1-t^q)^n(1-t)^{-(n+1)}\right)$;)

$$D_{XL} = \min\left\{D : [t^D]\ \left(\frac{(1-t^q)^n}{(1-t)^{n+1}}\ \prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right) \le 0\right\}.$$

$$D_{reg} = \min\left\{D : [t^D]\ \left(\frac{(1-t^q)^n}{(1-t)^{n}}\ \prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right)\right) < 0\right\}.$$

Large Fields ($q > D$, $T = \binom{n+D}{D}$)

$$D_{XL} = \min\left\{D : [t^D]\ \left((1-t)^{m-n-1}\,(1+t)^m\right) \le 0\right\},$$

$$D_{reg} = \min\left\{D : [t^D]\ \left((1-t)^{m-n}\,(1+t)^m\right) < 0\right\}.$$

$\mathbb{F}_2$ case ($T = \sum_{i=0}^{D}\binom{n}{i}$)

$$D_{XL} = \min\left\{D : [t^D]\ \left((1-t)^{-1}\,(1+t)^n\,(1+t^2)^{-m}\right) \le 0\right\},$$

$$D_{reg} = \min\left\{D : [t^D]\ \left((1+t)^n\,(1+t^2)^{-m}\right) < 0\right\}.$$

The Curious History of $\mathbb{F}_2$ with $m = n$

- Courtois+Pieprzyk overclaimed efficiency of XL in 2002
- Bardet et al 2004: derives $D = 0.0090 + o(1))n$ for $\mathbf{F_5}$.
- Yang and Chen 2004: independentaly finds sparse matrix solver with XL gets down to $2^{(0.873+o(1))n}$, with a similar asymptotic D.
- Bardet et al 2012: discusses a solver *which is exactly the same as XL with sparse matrix solver*, but improved with *guessing*.

Reasons to go back to XL

- Most of the time where it matters $D_{XL} - D_{reg} \leq 1$, this is true when $m/n = c + o(1)$ for most fields.
- XL with a sparse matrix solver has a smaller footprint. MAGMA $\mathbf{F_4}$ takes about 60GB of main memory when $m = n = 32$. XL much less.
- Asymptotically, if sparse matrix solving works, it would be faster than any dense-matrix elimination.

Guessing (Fixing) as an Optimization

- Courtois et al 2000: guessing at some variables may help
- Yang et al 2004: in random systems, the asymptotic proportion of resulting equations to variables depends only on the field size. The result applies equally to XL and $\mathbf{F_4}/\mathbf{F_5}$.

Guessing (Fixing) as an Optimization

- Courtois et al 2000: guessing at some variables may help
- Yang et al 2004: in random systems, the asymptotic proportion of resulting equations to variables depends only on the field size. The result applies equally to XL and $\mathbf{F_4}/\mathbf{F_5}$.
- Yang et al notes that for $m = n$ in $\mathbb{F}_2$, one should guess approximately $(0.45 + o(1))n$ variables, resulting in a complexity of $2^{(0.785+o(1))n}$ using sparse solvers

Guessing (Fixing) as an Optimization

- Courtois et al 2000: guessing at some variables may help
- Yang et al 2004: in random systems, the asymptotic proportion of resulting equations to variables depends only on the field size. The result applies equally to XL and $\mathbf{F_4}/\mathbf{F_5}$.
- Yang et al notes that for $m = n$ in $\mathbb{F}_2$, one should guess approximately $(0.45 + o(1))n$ variables, resulting in a complexity of $\underline{2^{(0.785+o(1))n}}$ using sparse solvers — they were too stupid to use Maple correctly

Guessing (Fixing) as an Optimization

- Courtois et al 2000: guessing at some variables may help
- Yang et al 2004: in random systems, the asymptotic proportion of resulting equations to variables depends only on the field size. The result applies equally to XL and $\mathbf{F_4}/\mathbf{F_5}$.
- Yang et al notes that for $m = n$ in $\mathbb{F}_2$, one should guess approximately $(0.45 + o(1))n$ variables, resulting in a complexity of $2^{(0.785+o(1))n}$ using sparse solvers — they were too stupid to use Maple correctly
- Bettale et al 2008: rediscovers this fact in $\mathbf{F_5}$, calls it "Hybrid $\mathbf{F_5}$", and applies it to both large and small fields.

Guessing (Fixing) as an Optimization

- Courtois et al 2000: guessing at some variables may help
- Yang et al 2004: in random systems, the asymptotic proportion of resulting equations to variables depends only on the field size. The result applies equally to XL and $\mathbf{F_4}/\mathbf{F_5}$.
- Yang et al notes that for $m = n$ in $\mathbb{F}_2$, one should guess approximately $(0.45 + o(1))n$ variables, resulting in a complexity of $2^{(0.785+o(1))n}$ using sparse solvers — they were too stupid to use Maple correctly
- Bettale et al 2008: rediscovers this fact in $\mathbf{F_5}$, calls it "Hybrid $\mathbf{F_5}$", and applies it to both large and small fields.
- Bardet et al 2012: correctly arrives at $2^{(0.791+o(1))n}$ at exactly the same $(0.45 + o(1))n$ variables guessed, using Sparse XL, and predicts $n = 200$ as crossover point against brute-force.

True Cost of Gröbner Bases vs. Brute Force

The rationale predicting crossover at $m = n = 200$

Guessing 130 variables, we can solve the system in 2^{197} field operations, with each system of $(m, n) = (200, 70)$ taking 2^{67} field operations. Each subsystem is a sparse in $2^{30.4}$ $\mathbb{F}_2$-variables and $2^{44.8}$ bytes of storage.

True Cost of Gröbner Bases vs. Brute Force

The rationale predicting crossover at $m = n = 200$

Guessing 130 variables, we can solve the system in 2^{197} field operations, with each system of $(m, n) = (200, 70)$ taking 2^{67} field operations. Each subsystem is a sparse in $2^{30.4}$ $\mathbb{F}_2$-variables and $2^{44.8}$ bytes of storage.

Memory Access Problem

No way a field operation in Wiedemann is as efficient as one in brute-force attack — each AND is preceded by an essentially random memory read.

True Cost of Gröbner Bases vs. Brute Force

The rationale predicting crossover at $m = n = 200$

Guessing 130 variables, we can solve the system in 2^{197} field operations, with each system of $(m, n) = (200, 70)$ taking 2^{67} field operations. Each subsystem is a sparse in $2^{30.4}$ $\mathbb{F}_2$-variables and $2^{44.8}$ bytes of storage.

Memory Access Problem

No way a field operation in Wiedemann is as efficient as one in brute-force attack — each AND is preceded by an essentially random memory read.

Equipment Scaling Problem at only 2^{39} bytes

- Current quote of 512GB server from a reputable vendor (Tyan) is 2U at US$13,000, 4 Opteron "Interlagos" 6282 (2.6GHz 16 cores).
- Can get "TWIN2" 4 servers in 2U with *eight* such Opterons, and 2GB of RAM per core at the same price from the same vendor.
- 32 desktops each with 8 opteron 2.6GHz cores at same price.

True Cost of Gröbner Bases vs. Brute Force

Bouillaguet et al 2010: brute force attacks on $\mathbb{F}_2$ run very efficiently on nice toys like GPUs on video cards, and it is embarassingly parallel.

True Cost of Gröbner Bases vs. Brute Force

Bouillaguet et al 2010: brute force attacks on $\mathbb{F}_2$ run very efficiently on nice toys like GPUs on video cards, and it is embarassingly parallel.

Quote from Daniel J. Bernstein

To increase total memory by $M\times$, one should be expected to incur a per-access cost of $\sqrt{M}\times$.

True Cost of Gröbner Bases vs. Brute Force

Bouillaguet et al 2010: brute force attacks on $\mathbb{F}_2$ run very efficiently on nice toys like GPUs on video cards, and it is embarassingly parallel.

Quote from Daniel J. Bernstein

To increase total memory by $M\times$, one should be expected to incur a per-access cost of $\sqrt{M}\times$.

Quote from local friendly system vendor

To increase total memory in a high-speed interconnected cluster by $M\times$ incurs a unit cost increase penalty of about $\sqrt[3]{M}\times$.

True Cost of Gröbner Bases vs. Brute Force

Bouillaguet et al 2010: brute force attacks on $\mathbb{F}_2$ run very efficiently on nice toys like GPUs on video cards, and it is embarassingly parallel.

Quote from Daniel J. Bernstein

To increase total memory by $M\times$, one should be expected to incur a per-access cost of $\sqrt{M}\times$.

Quote from local friendly system vendor

To increase total memory in a high-speed interconnected cluster by $M\times$ incurs a unit cost increase penalty of about $\sqrt[3]{M}\times$.

Our Guesstimates

With $\omega = 2.5$ (much less $\omega = 2.83$) as the "real cost" exponent of matrix solving, the crossover point of XL with Sparse Solver vs Brute Force will not happen until $n \approx 650$, which should be long past any point of relevancy for cryptographic considerations.

Asymptotics by Contour Integrals

The degree d coefficient of the Maclaurin series of $f(t)$ is given by a contour integral $S_f(d) := (2\pi i)^{-1} \oint \left(f(z)\, z^{-(d+1)} \right) dz$. The power of the first nonpositive (or resp. negative) coefficient of $f(t)$ is then the smallest integer no less (resp. greater) than the smallest positive real root of S_f. Here we will denote by $\widehat{D_{XL}}$ and $\widehat{D_{reg}}$ the smallest positive roots of the corresponding integral functions, hence $D_{XL} = \lceil \widehat{D_{XL}} \rceil$, $D_{reg} = \lfloor \widehat{D_{reg}} \rfloor + 1$. Using Saddle Point methods we can get

Asymptotics by Contour Integrals

The degree d coefficient of the Maclaurin series of $f(t)$ is given by a contour integral $S_f(d) := (2\pi i)^{-1} \oint \left(f(z)\, z^{-(d+1)}\right) dz$. The power of the first nonpositive (or resp. negative) coefficient of $f(t)$ is then the smallest integer no less (resp. greater) than the smallest positive real root of S_f. Here we will denote by $\widehat{D_{XL}}$ and $\widehat{D_{reg}}$ the smallest positive roots of the corresponding integral functions, hence $D_{XL} = \lceil \widehat{D_{XL}} \rceil$, $D_{reg} = \lfloor \widehat{D_{reg}} \rfloor + 1$. Using Saddle Point methods we can get

Barely Overdetermined ($m/n = 1 + o(1)$) Large Fields

Empirically $D_{XL} - D_{reg} > 0$. If $m, n \to \infty$ while $m - n = f > 1$ fixed, D_{reg} is asymptotically $\widehat{D_{reg}} = \frac{m}{2} - h_{f,1} \cdot \sqrt{\frac{m}{2}} \cdot (1 + o(1))$. So $D_{XL} - D_{reg} = (h_{f+1,1} - h_{f,1})\sqrt{\frac{m}{2}}$ is large.

This explains why XL takes a long time catching up to $\mathbf{F_4}$ when $m \approx n$.

Upshot of asymptotics: $\widehat{D_{XL}} - \widehat{D_{reg}} = 0.207$ for $m = 2n$?

Asymptotics for large q, $m/n = \alpha + o(1)$

D_{reg} is approximated by the Coalescent Saddle Point Method

$$\widehat{D_{reg}} = \left(\alpha - \frac{1}{2} - \sqrt{\alpha(\alpha-1)}\right) n + \frac{-a_1}{2\sqrt[6]{\alpha(\alpha-1)}} n^{\frac{1}{3}} - \left(2 - \frac{2\alpha-1}{4\sqrt{\alpha(\alpha-1)}}\right) + O(n^{-\frac{1}{3}}).$$

Upshot of asymptotics: $\widehat{D_{XL}} - \widehat{D_{reg}} = 0.207$ for $m = 2n$?

Asymptotics for large q, $m/n = \alpha + o(1)$

D_{reg} is approximated by the Coalescent Saddle Point Method

$$\widehat{D_{reg}} = \left(\alpha - \frac{1}{2} - \sqrt{\alpha(\alpha-1)}\right) n + \frac{-a_1}{2\sqrt[6]{\alpha(\alpha-1)}} n^{\frac{1}{3}} - \left(2 - \frac{2\alpha-1}{4\sqrt{\alpha(\alpha-1)}}\right) + O(n^{-\frac{1}{3}}).$$

$m = 2n\,(\alpha = 2)$ case

Coalescent Saddles computations gets us

$$\widehat{D_{reg}}(n, 2n) = 0.0858n + 1.0415n^{1/3} - 1.4697 + O(n^{-1/3}).$$

But, we can carry through the same computations to find

$$\begin{aligned} \widehat{D_{XL}}(n, 2n) &= 0.0858n + 1.0415n^{1/3} - 1.2626 + O(n^{-1/3}). \\ &= D_{reg}(n, 2n) + 0.2071 + o(1). \end{aligned}$$

Heuristic Proof

We can write the uniform asymptotic expansion as follows

$$\widehat{D_{reg}} = \left(1 - \frac{\alpha^{-1}}{2} - \sqrt{(1-\alpha^{-1})}\right) m + \frac{-a_1 \cdot (\alpha^{-1})^{2/3}}{2(1-\alpha^{-1}))^{\frac{1}{6}}} m^{\frac{1}{3}} - \left(2 - \frac{2-\alpha^{-1}}{4(1-\alpha^{-1})^{\frac{1}{2}}}\right) + O(\frac{1}{m^{1/3}}).$$

So if we write $\widehat{D_{reg}}(n, \alpha n) = f(\alpha^{-1}, m)$, then

$$\widehat{D_{XL}}(n,2n) - \widehat{D_{reg}}(n,2n) = f\left(\left(\tfrac{1}{2} + \tfrac{1}{2n}\right), 2n\right) - f\left(\tfrac{1}{2}, 2n\right) \approx \frac{1}{2n} \cdot \frac{\partial f}{\partial(\alpha^{-1})}\Big|_{\alpha^{-1}=\frac{1}{2}, m=2n} = \left(\frac{\sqrt{2}-1}{2}\right) + o(1)$$

Heuristic Proof

We can write the uniform asymptotic expansion as follows

$$\widehat{D_{reg}} = \left(1 - \frac{\alpha^{-1}}{2} - \sqrt{(1-\alpha^{-1})}\right) m + \frac{-a_1 \cdot (\alpha^{-1})^{2/3}}{2(1-\alpha^{-1}))^{\frac{1}{6}}} m^{\frac{1}{3}} - \left(2 - \frac{2-\alpha^{-1}}{4(1-\alpha^{-1})^{\frac{1}{2}}}\right) + O(\frac{1}{m^{1/3}}).$$

So if we write $\widehat{D_{reg}}(n, \alpha n) = f(\alpha^{-1}, m)$, then

$$\widehat{D_{XL}}(n,2n) - \widehat{D_{reg}}(n,2n) = f\left(\left(\tfrac{1}{2} + \tfrac{1}{2n}\right), 2n\right) - f\left(\tfrac{1}{2}, 2n\right) \approx \frac{1}{2n} \cdot \frac{\partial f}{\partial(\alpha^{-1})}\Big|_{\alpha^{-1}=\frac{1}{2}, m=2n} = \left(\frac{\sqrt{2}-1}{2}\right) + o(1)$$

Empirically tested

This explains why $D_{XL}(n, 2n+k) - D_{reg}(n, 2n+k)$ is on average 0.207 for $k = -10, \ldots, 10$. Similarly, we can verify that for $m/n \approx 1.5$ and 2.5, the XL to $\mathbf{F_4}$ degree drop converges to 0.367 and 0.145, respectively.

For many QUAD-like generic systems of cryptographic significance, we have shown that $D_{XL} - D_{reg}$ is small, which should imply that XL with a sparse solver would eventually be better than $\mathbf{F_4}$, especially given True Cost considerations.

Take-away Points (1)

The Power of Brute Force

We estimate Gröbner Bases type methods should not be used to estimate the complexity of solving generic $\mathbb{F}_2$ $\mathcal{MQ}$ systems up to about $m = 500$. One should simply assume brute force.

Take-away Points (1)

The Power of Brute Force

We estimate Gröbner Bases type methods should not be used to estimate the complexity of solving generic $\mathbb{F}_2$ $\mathcal{MQ}$ systems up to about $m = 500$. One should simply assume brute force.

Higher Order Polynomials

One can do cubic and quartic generic systems and in these cases we should definitely go with brute force because Gröbner bases methods have little hope.

Take-Away Points (2)

In many cryptographically interesting cases the degree of operations for XL and $\mathbf{F_4}/\mathbf{F_5}$ remains very close. In these situations, XL with a sparse matrix solver will eventually be better than $\mathbf{F_4}/\mathbf{F_5}$ unless the number of columns and rows in the latter can be pared down sufficiently.

Take-Away Points (2)

In many cryptographically interesting cases the degree of operations for XL and $\mathbf{F_4}/\mathbf{F_5}$ remains very close. In these situations, XL with a sparse matrix solver will eventually be better than $\mathbf{F_4}/\mathbf{F_5}$ unless the number of columns and rows in the latter can be pared down sufficiently.

Estimate of the number of Columns and Rows

Articles on $\mathbf{F_4}/\mathbf{F_5}$ and similar algorithms often uses the same T as we showed before to estimate the runtime. We should investigate more carefully into how many monomials never appear in the higher degrees.

Future Work

- We need to investigate the distribution of non-zero entries in a typical $\mathbf{F_4}/\mathbf{F_5}$ run to make a real comparison.
- Chou et al at CHES 2012 demonstrate that XL can be implemented well in parallel (such that it is likely better than $\mathbf{F_4}$ on a per-GHz basis for some isolated cases), and there are similar work going on for $\mathbf{F_4}/\mathbf{F_5}$. Good implementations needs to be made available for the general public.
- For non-generic systems, XL does not do well. However, there are cases where one could run XL2 with a sparse matrix solver as a variant of $\mathbf{F_4}$. We have demonstrated that it works and in some isolated cases can be the most efficient solver. More exploration is called for in this area.

Thanks!

- Paper will be available on ePrint (soon)
- Questions?