A SAT-Based Approach for Index Calculus on Binary Elliptic Curves

Monika Trimoska   Sorina Ionica   Gilles Dequen

MIS Laboratory, University of Picardie Jules Verne

AfricaCrypt 2020

Discrete log

Defining discrete log

Given a finite cyclic group $(G, +)$ and two elements $g, h \in G$, find $x \in \mathbb{Z}$ such that

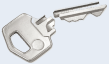
$$h = x \cdot g.$$

Generic attacks

Pollard rho, Baby-step Giant-step, Kangaroo

Index calculus attack

Subexponential in $((\mathbb{Z}/p\mathbb{Z})^*, \cdot)$.



Index calculus on elliptic curves

Let $\mathbb{F}_{2^n}$ be a finite field and E be an elliptic curve defined by

$$\boxed{E : y^2 + xy = x^3 + ax^2 + b}$$

with $a, b \in \mathbb{F}_{2^n}$.

Discrete log:

Find x, such that $xP = Q$, where $P, Q \in E(F_{2^n})$.

Point decomposition phase of the Index calculus algorithm:

Find $P_1, \ldots, P_{m-1} \in E(\mathbb{F}_{2^n})$, such that

$$P_m = P_1 + \ldots + P_{m-1}$$

Point Decomposition Problem (PDP)

Semaev's summation polynomials (2004)

$$S_2(X_1, X_2) = X_1 + X_2,$$
$$S_3(X_1, X_2, X_3) = X_1^2X_2^2 + X_1^2X_3^2 + X_1X_2X_3 + X_2^2X_3^2 + b,$$

For $m \geq 4$

$$S_m(X_1, \ldots, X_m) =$$
$$Res_X(S_{m-k}(X_1, \ldots, X_{m-k-1}, X), S_{k+2}(X_{m-k}, \ldots, X_m, X))$$

For $P_1, \ldots, P_m \in E(\mathbb{F}_{2^n})$

$$P_1 + \ldots + P_m = \mathcal{O} \iff S_m(\mathbf{x}_{P_1}, \ldots, \mathbf{x}_{P_m}) = 0$$

Gaudry and Diem (2008 and 2009)

Weil descent

Rewrite the equation $S_m(X_1, \ldots, X_m) = 0$ as a system of n equations over $\mathbb{F}_2$.

Example (trivial case of $m = 2$):

$S_2(X_1, X_2) = 0$

$X_1 + X_2 = 0$

$(a_{1,0} + a_{1,1}t + \ldots + a_{1,n-1}t^{n-1}) + (a_{2,0} + a_{2,1}t + \ldots + a_{2,n-1}t^{n-1}) = 0$

$(a_{1,0} + a_{2,0}) + (a_{1,1} + a_{2,1})t + \ldots + (a_{1,n-1} + a_{2,n-1})t^{n-1} = 0$

$$\begin{cases} a_{1,0} + a_{2,0} = 0 \\ a_{1,1} + a_{2,1} = 0 \\ \ldots \\ a_{1,n-1} + a_{2,n-1} = 0 \end{cases}$$

Gaudry and Diem (2008 and 2009)

Symmetrization

Rewrite S_m in terms of the elementary symmetric polynomials

$$\begin{aligned}
\mathbf{e}_1 &= \sum_{1\leq i_1\leq m} X_{i_1}, \\
\mathbf{e}_2 &= \sum_{1\leq i_1,i_2\leq m} X_{i_1}X_{i_2}, \\
&\dots \\
\mathbf{e}_m &= \prod_{1\leq i\leq m} X_i.
\end{aligned}$$

PDP algebraic model

Choice of a factor base : an l-dimensional vector subspace V of $\mathbb{F}_{2^n}/\mathbb{F}_2$. When $l \sim \frac{n}{m}$ the system has a reasonable chance to have a solution.

X_i-variables

$$X_1 = a_{1,0} + \ldots + a_{1,l-1} t^{l-1}$$

$$X_2 = a_{2,0} + \ldots + a_{2,l-1} t^{l-1}$$

$$\ldots$$

$$X_m = a_{m,0} + \ldots + a_{m,l-1} t^{l-1}$$

$\mathbf{e}_i$-variables

$$\mathbf{e}_1 = e_{1,0} + \ldots + e_{1,l-1} t^{l-1}$$

$$\mathbf{e}_2 = e_{2,0} + \ldots + e_{2,2l-2} t^{2l-2}$$

$$\ldots$$

$$\mathbf{e}_m = e_{m,0} + \ldots + e_{m,m(l-1)} t^{m(l-1)}$$

PDP algebraic model

Two sets of equations

- Equations defining symmetric polynomials

$$
\begin{aligned}
e_{1,0} &= a_{1,0} + \ldots + a_{m,0} \\
e_{1,1} &= a_{1,1} + \ldots + a_{m,1} \\
\ldots \\
e_{m,m(l-1)} &= a_{1,l} \cdot \ldots \cdot a_{m,l}.
\end{aligned}
$$

- Equations derived from the Weil descent

The system is commonly solved using Gröbner basis methods.

Algebraic model to SAT-reasoning model

Using SAT solvers as a cryptanalytic tool requires expressing the cryptographic problem as a Boolean formula in conjunctive normal form (CNF) - a conjunction ($\wedge$) of OR-clauses.

Example.

$$
\begin{array}{l}
(\neg x_1 \vee x_2) \wedge \\
(\neg x_2 \vee x_4 \vee \neg x_5)) \wedge \\
(x_5 \vee x_6)
\end{array}
$$

Algebraic model to SAT-reasoning model

XOR-enabled SAT solvers are adapted to read a formula in CNF-XOR form - a conjunction ($\wedge$) of OR-clauses and XOR-clauses.

Example.

$$
\begin{aligned}
&(\neg x_1 \vee x_2) \wedge \\
&(\neg x_2 \vee x_4 \vee \neg x_5)) \wedge \\
&(x_1 \oplus x_5 \oplus x_6)
\end{aligned}
$$

Algebraic model to CNF-XOR model

Variables in $\mathbb{F}_2$:
$\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6.$

Propositional variables:
$x_1, x_2, x_3, x_4, x_5, x_6$ with truth values in $\{\text{TRUE}, \text{FALSE}\}$

$$\begin{aligned}
&\mathbf{x}_1 + \mathbf{x}_2 \cdot \mathbf{x}_4 + \mathbf{x}_5 \cdot \mathbf{x}_6 + 1 = 0\\
&\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_4 + \mathbf{x}_5 + 1 = 0\\
&\mathbf{x}_3 + \mathbf{x}_4 + \mathbf{x}_2 \cdot \mathbf{x}_4 + 1 = 0\\
&\mathbf{x}_2 + \mathbf{x}_5 + \mathbf{x}_2 \cdot \mathbf{x}_4 + \mathbf{x}_5 \cdot \mathbf{x}_6 + 1 = 0\\
&\mathbf{x}_3 + \mathbf{x}_4 + \mathbf{x}_6 + 1 = 0
\end{aligned}$$

$$\longrightarrow$$

$$\begin{aligned}
&(x_1 \oplus (x_2 \wedge x_4) \oplus (x_5 \wedge x_6)) \wedge\\
&(x_1 \oplus x_2 \oplus x_4 \oplus x_5) \wedge\\
&(x_3 \oplus x_4 \oplus (x_2 \wedge x_4)) \wedge\\
&(x_2 \oplus x_5 \oplus (x_2 \wedge x_4) \oplus (x_5 \wedge x_6)) \wedge\\
&(x_3 \oplus x_4 \oplus x_6)
\end{aligned}$$

Multiplication in $\mathbb{F}_2$ $(\cdot)$ becomes the logical AND operation $(\wedge)$ and addition in $\mathbb{F}_2$ $(+)$ becomes the logical XOR $(\oplus)$.

Algebraic model to CNF-XOR model

Add new variable x_7 to substitute the conjunction $x_2 \wedge x_4$. We have that

$$\begin{tikzcd}
 & & & x_7 \Leftrightarrow (x_2 \wedge x_4) \arrow[d] & \\
 & & & (x_7 \Rightarrow (x_2 \wedge x_4)) \wedge ((x_2 \wedge x_4) \Rightarrow x_7) \arrow[dll] \arrow[dr] & \\
 & \neg x_7 \vee (x_2 \wedge x_4) \arrow[dl] \arrow[dr] & & & \neg(x_2 \wedge x_4) \vee x_7 \arrow[d] \\
(\neg x_7 \vee x_2) & \wedge & (\neg x_7 \vee x_4) & & \neg x_2 \vee \neg x_4 \vee x_7
\end{tikzcd}$$

Algebraic model to CNF-XOR model

Propositional variables:

$x_1, x_2, x_3, x_4, x_5, x_6$ with truth values in $\{\text{TRUE}, \text{FALSE}\}$

$$
\begin{aligned}
&(x_1 \oplus (x_2 \wedge x_4) \oplus (x_5 \wedge x_6)) \wedge \\
&(x_1 \oplus x_2 \oplus x_4 \oplus x_5) \wedge \\
&(x_3 \oplus x_4 \oplus (x_2 \wedge x_4)) \wedge \\
&(x_2 \oplus x_5 \oplus (x_2 \wedge x_4) \oplus (x_5 \wedge x_6)) \wedge \\
&(x_3 \oplus x_4 \oplus x_6)
\end{aligned}
$$

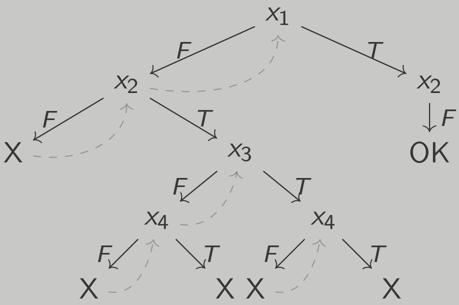
$$\longrightarrow$$

$$
\begin{aligned}
&(\neg x_7 \vee x_2) \wedge \\
&(\neg x_7 \vee x_4) \wedge \\
&(\neg x_2 \vee \neg x_4 \vee x_7) \wedge \\
&(\neg x_8 \vee x_5) \wedge \\
&(\neg x_8 \vee x_6) \wedge \\
&(\neg x_5 \vee \neg x_6 \vee x_8) \wedge \\
&(x_1 \oplus x_7 \oplus x_8) \wedge \\
&(x_1 \oplus x_2 \oplus x_4 \oplus x_5) \wedge \\
&(x_3 \oplus x_4 \oplus x_7) \wedge \\
&(x_2 \oplus x_5 \oplus x_7 \oplus x_8) \wedge \\
&(x_3 \oplus x_4 \oplus x_6)
\end{aligned}
$$

WDSat algorithm

Based on the Davis-Putnam-Logemann-Loveland (DPLL) algorithm.

Recursively building a binary search-tree of height equivalent (at worst) to the number of variables.



WDSat - Three reasoning modules

CNF module

Performs unit propagation on CNF-clauses.

XORSET module

Performs unit propagation on the parity constraints. When all except one literal in a XOR clause is assigned, we infer the truth value of the last literal according to parity reasoning.

XORGAUSS module

Performs Gaussian elimination on the XOR system.

WDSat - breaking symmetry

- Exploiting the symmetry of Semaev's summation polynomials: when $X_1, ..., X_m$ is a solution, all permutations of this set are a solution as well.
- Establish the following constraint $X_1 \leq X_2 \leq \ldots \leq X_m$.
- Implement constraint in the solver using a tree-pruning-like technique.
- Optimizes the complexity by a factor of $m!$.

Experimental results

			SATisfiable			UNSATisfiable		
Approach	l	n	Runtime	#Conflicts	Memory	Runtime	#Conflicts	Memory
Gröbner basis	6	17	207.220	NA	3601	142.119	NA	3291
		19	215.187	NA	3940	155.765	NA	4091
	7	19	3854.708	NA	38763	2650.696	NA	38408
		23	3128.844	NA	35203	2286.136	NA	35162
CryptoMiniSat	6	17	15.673	61812	34.5	62.396	260843	39.3
		19	14.128	53767	33.2	64.563	259688	42.1
	7	19	176.463	484098	41.5	843.367	2077747	72.3
		23	300.021	638152	48.9	1012.412	2070190	73.6
WDSAT	6	17	.601	49117	1.4	3.851	254686	1.4
		19	.470	38137	1.4	3.913	255491	1.4
	7	19	9.643	534867	16.7	44.107	2073089	16.7
		23	9.303	477632	16.7	47.347	2067168	16.7
WDSAT+br-sym	6	17	.220	17792	1.4	.605	43875	1.4
		19	.243	19166	1.4	.639	44034	1.4
	7	19	2.205	130062	1.4	6.859	351353	1.4
		23	3.555	189940	1.4	7.478	350257	1.4

Table: Comparing Gröbner basis and SAT-based approaches for solving the PDP. Running times are in seconds and memory is in MB.

Experimental results

		SATisfiable			UNSATisfiable		
l	n	Runtime	#Conflicts	Memory	Runtime	#Conflicts	Memory
8	23	29.584	1145966	17.0	81.767	2800335	17.0
9	37	447	10557129	17.1	1048	22396994	17.1
	47	609	12675174	17.2	1167	22381494	17.2
	59	611	11297325	17.3	1327	22390211	17.3
	67	677	11608420	17.4	1430	22388053	17.4
10	47	5847	95131900	17.3	11963	179019409	17.3
	59	6849	97254458	17.4	13649	179067171	17.4
	67	6530	88292215	17.4	14555	179052277	17.4
	79	7221	86174432	17.5	16294	179043408	17.5
11	59	64162	727241718	19.2	135801	1432191354	19.2
	67	70075	741222864	19.3	145357	1432183842	19.3
	79	61370	599263451	19.4	161388	1432120827	19.4
	89	85834	736610196	19.5	175718	1432099666	19.5

Table: Experimental results using the WDSat solver with breaking symmetry. Running times are in seconds and memory is in MB.

Conclusion

- When solving the PDP for prime degree extension fields $\mathbb{F}_2$, Gröbner basis methods can be replaced with a SAT-based approach.
- Our CNF-XOR model with the dedicated SAT-solver, WDSAT, yields significantly faster running times than all other algebraic and SAT-based approaches.
- The memory is no longer a constraint for the PDP.
- Extending the WDSAT solver with our symmetry breaking technique optimizes the resolution of the PDP by a factor of m!.