Algorithms for the Approximate Common Divisor Problem

Steven D. Galbraith, Shishay W. Gebregiyorgis and Sean Murphy
University of Auckland and Royal Holloway
Thanks

- Referees and Program Committee
- Nadia Heninger
- Tancrède Lepoint
Outline

- Approximate common divisor problem (ACD).
- Simultaneous Diophantine approximation.
- Orthogonal lattice method.
- Multivariate polynomial approach.
- Main conclusion: multivariate polynomial approach is not better than the other lattice methods for practical cryptanalysis.
- Sample-amplification and pre-processing approaches.
- Open problems.
Approximate Common Divisor problem (ACD)

- Introduced by Howgrave-Graham.
- Given $x_i = pq_i + r_i$ with $|r_i| \ll p$ for $1 \leq i \leq t$ to compute $p$.
- This is a well-defined problem if one is given enough samples.
Van Dijk, Gentry, Halevi and Vaikuntanathan proposed a homomorphic encryption scheme based on ACD.

Ciphertexts are $c = pq + 2r + m$ where $m \in \{0, 1\}$ is message and $|r| \ll p$.

To decrypt: reduce modulo $p$ and then modulo 2.

Homomorphic for addition:

$$c_1 + c_2 = p(q_1 + q_2) + 2(r_1 + r_2) + (m_1 + m_2)$$

decysts to $m_1 + m_2 \pmod{2}$.

Homomorphic for multiplication:

$$c_1 c_2 = p(pq_1 q_2 + 2q_1 r_2 + 2q_2 r_1) + 2(2r_1 r_2 + r_1 m_2 + r_2 m_1) + (m_1 m_2)$$

which decrypts to $m_1 m_2 \pmod{2}$ as long as $2r_1 r_2 \ll p$. 

Further variants


Cheon and Stehlé variant

- New harder variant of the problem: If LWE hard then ACD hard.
- More efficient homomorphic encryption using “scale invariant” concept.
Formal ACD problem

- Fix $\gamma, \eta, \rho \in \mathbb{N}$ with $\gamma > \eta > \rho$.
- $p$ is an $\eta$-bit odd integer.
- Define
  \[ D_{\gamma, \rho}(p) = \{ pq + r \mid q \leftarrow \mathbb{Z} \cap [0, 2^{\gamma}/p), r \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho) \}. \]

- **Approximate common divisor problem (ACD):** Given polynomially many samples $x_i$ from $D_{\gamma, \rho}(p)$, to compute $p$.
- **Partial approximate common divisor problem (PACD):** Given polynomially many samples $x_i$ from $D_{\gamma, \rho}(p)$ and also a sample $x_0 = pq_0$ for uniformly chosen $q_0 \in \mathbb{Z} \cap [0, 2^{\gamma}/p)$, to compute $p$.
- There are also “decisional” versions.
Parameters

- Let $\lambda$ be a security parameter.
- Take $\rho = \lambda$ due to attacks on the term $r$ in $pq + r$. See Chen-Nguyen, Coron-Naccache-Tibouchi, Lee-Seo.
- Van Dijk et al set $\gamma/\eta^2 = \omega(\log(\lambda))$ to thwart lattice attacks on the approximate common divisor problem.
- Suggested parameters $(\rho, \eta, \gamma) = (\lambda, \lambda^2, \lambda^5)$
- One example $(\rho, \eta, \gamma) = (71, 2698, 19350000)$. Yes, each ACD sample $x_i = pq_i + r_i$ is 19 million bits (about 2.4 megabytes).
Variants

- **CRT-ACD problem**
  - Cheon et al set \( \pi = p_1 \cdots p_{\ell} \) and \( x_0 = \pi q_0 \).
  - A ciphertext is \( c = \pi q + r \equiv 2r + m_i \pmod{p_i} \) for all \( i \).
  - Problem is to compute \( p_1, \ldots, p_{\ell} \).
  - It is an open problem to give an algorithm to solve the CRT-ACD problem that exploits the CRT structure.

- **Cheon-Stehlé approximate common divisor problem**
  - Parameters \((\rho, \eta, \gamma) = (\lambda, \lambda + d \log(\lambda), \Omega(d^2 \lambda \log(\lambda)))\),
  - where \( d \) is the homomorphic circuit depth.
  - Note that \( \rho \) is no longer extremely small compared with \( \eta \).
Simultaneous Diophantine approximation approach (SDA)

- Due to Howgrave-Graham.
- Does not benefit from having an exact sample \( x_0 = pq_0 \), so suppose \( x_0 = pq_0 + r_0 \).
- If \( x_i = pq_i + r_i \) for \( 1 \leq i \leq t \), where \( r_i \) is small, then

\[
\frac{x_i}{x_0} \approx \frac{q_i}{q_0}
\]

for \( 1 \leq i \leq t \).
- In other words, the fractions \( q_i/q_0 \) are an instance of simultaneous Diophantine approximation to \( x_i/x_0 \).
Simultaneous Diophantine approximation approach (SDA)

Define lattice $L$ of rank $t + 1$ with (row) basis

$$B = \begin{pmatrix} 2^{\rho+1} & x_1 & x_2 & \cdots & x_t \\ -x_0 & -x_0 & \ddots & \ddots \\ & & & -x_0 \end{pmatrix}.$$  

Note $\det(L) = 2^{\rho+1} x_0^t$.

Note that $L$ contains the vector

$$v = (q_0, q_1, \cdots, q_t)B$$

$$= (2^{\rho+1} q_0, q_0 x_1 - q_1 x_0, \cdots, q_0 x_t - q_t x_0)$$

$$= (q_0 2^{\rho+1}, q_0 r_1 - q_1 r_0, \cdots, q_0 r_t - q_t r_0).$$
SDA algorithm

If

\[ \|v\| \approx \sqrt{t + 1} 2^{\gamma - \eta + \rho + 1} < \sqrt{\frac{t + 1}{2\pi e}} \det(L)^{1/(t+1)} \]

then we expect target vector \( v \) to be the shortest non-zero vector in the lattice.

The attack is to run a lattice basis reduction algorithm to get a candidate \( w \) for the shortest non-zero vector.

One then divides the first entry of \( w \) by \( 2^{\rho + 1} \) to get a candidate value for \( q_0 \) and then computes \( r_0 = x_0 \pmod{q_0} \) and \( p = (x_0 - r_0)/q_0 \).

One can then “test” this value for \( p \) by checking if \( x_i \pmod{p} \) are small for all \( 1 \leq i \leq t \).
Remarks

- Attack only requires a single short vector, not a large number of short vectors.
- Analysis of the attack is heuristic.
- To use LLL, need target $\mathbf{v}$ to be shorter by an exponential factor than the second successive minimum. So need

$$2^{t/2} \|\mathbf{v}\| \leq \sqrt{n} \det(L)^{1/(t+1)}.$$ 

- Necessary condition for algorithm to succeed is

$$t + 1 > \frac{\gamma - \rho}{\eta - \rho}.$$ 

- Consistent with work of Cheon-Stehlé.
- See paper for more details and discussion.
CRT case

- Have \( x_i = p_j q_{i,j} + r_{i,j} \) for \( 1 \leq j \leq \ell \) where each \( r_{i,j} \) is small.
- It follows that the lattice contains the vectors
  
  \[
  (q_{0,j} 2^{\rho+1}, q_{0,j} r_{1,j} - q_{1,j} r_{0,j}, \ldots, q_{0,j} r_{t,j} - q_{t,j} r_{0,j})
  \]

  for all \( 1 \leq j \leq \ell \) and these all have similar length.
- The \( j \)-th vector allows to compute \( p_j \).
- But any short linear combination of several of these vectors is also a short vector in the lattice, but not good for breaking the system.
Orthogonal Lattice Approach (OL)

- Nguyen and Stern promoted the orthogonal lattice for cryptanalysis.
- Appendix B.1 of van Dijk et al gives a method based on vectors orthogonal to \((x_1, \ldots, x_t)\). Their idea is that the lattice of integer vectors orthogonal to \((x_1, \ldots, x_t)\) contains the sublattice of integer vectors orthogonal to both \((q_1, \ldots, q_t)\) and \((r_1, \ldots, r_t)\).
- They also have a method based on vectors orthogonal to \((1, -r_1/R, \ldots, -r_t/R)\), where \(R = 2^\rho\).
- Ding and Tao have given a method based on vectors orthogonal to \((q_1, \ldots, q_t)\).
- Cheon and Stehlé considered the second method of DGHV.
- Our analysis and experiments suggest all these methods are essentially equivalent in both theory and practice.
Orthogonal Lattice Approach (OL)

- Need to have $t - 1$ linearly independent vectors in the lattice $L$ that satisfy a certain bound.
- Our approach is a bit simpler than previous works.
- We show that a necessary condition on the dimension is $t \geq (\gamma - \rho)/(\eta - \rho)$. Same as the SDA condition.
- In practice the OL method slightly faster than SDA as numbers smaller.
Multivariate polynomial approach (MP)

- Howgrave-Graham was the first to reduce the approximate common divisor problem to the problem of finding small roots of multivariate polynomial equations.
- The idea was further extended in Appendix B.2 of van Dijk et al.
- A detailed analysis was given by Cohn and Heninger in ANTS 2012.
- A variant for the case when the “errors” are not all the same size was given by Takayasu and Kunihiro.
- Cohn and Heninger show that this approach has advantages over the others if the number of ACD samples is very small (the original context studied by Howgrave-Graham).
- Our heuristic analysis and experimental results suggest that the multivariate approach has no advantage over the SDA or OL methods for practical cryptanalysis.
Multivariate polynomial approach (MP)

- Notation from Cohn and Heninger:
- Assume we have $N = pq_0$.
- Let $a_i = pq_i + r_i$ for $1 \leq i \leq m$ be ACD samples, where $|r_i| \leq R$ for some given bound $R$.
- Construct a polynomial $Q(X_1, X_2, \ldots, X_m)$ in $m$ variables such that $Q(r_1, \cdots, r_m) \equiv 0 \mod p^k$ for some $k$.
- Such polynomials are integer linear combinations of

$$
(X_1 - a_1)^{i_1} \cdots (X_m - a_m)^{i_m} N^\ell
$$

where $\ell$ is chosen such that $i_1 + \cdots + i_m + \ell \geq k$.
- An additional generality is to choose a degree bound $t \geq k$ and impose the condition $i_1 + \cdots + i_m \leq t$.
- The value $t$ will be optimised later.
- There is no benefit to taking $k > t$. 
Multivariate polynomial approach (MP)

- The lattice $L$ has dimension $d = \binom{t+m}{m}$ and determinant
  
  $$
  \det(L) = R\binom{t+m}{m} \frac{mt}{m+1} N\binom{k+m}{m} \frac{k}{m+1} = 2^d \frac{\rho mt}{m+1} + \binom{k+m}{m} \frac{\gamma k}{m+1}
  $$

  where we use the natural choice $R = 2^\rho$.

- Let $\mathbf{v}$ be a vector in $L$.

- One can interpret $\mathbf{v} = (v_{i_1, \ldots, i_m} R^{i_1+\cdots+i_m})$ as the coefficient vector of a polynomial

  $$
  Q(X_1, \ldots, X_m) = \sum_{i_1, \ldots, i_m} v_{i_1, \ldots, i_m} X_1^{i_1} \cdots X_m^{i_m}.
  $$
Multivariate polynomial approach (MP)

- So a short vector in $L$ gives a polynomial $Q$.
- If $|Q(r_1, \cdots, r_m)| < p^k$ then we have $Q(r_1, \cdots, r_m) = 0$ over the integers.

We have

$$|Q(r_1, \cdots, r_m)| \leq \sum_{i_1, \cdots, i_m} |v_{i_1 \cdots i_m}| r_1^{i_1} \cdots r_m^{i_m}$$

$$\leq \sum_{i_1, \cdots, i_m} |v_{i_1 \cdots i_m}| R_1^{i_1} \cdots R_m^{i_m}$$

$$= \|v\|_1.$$

- Hence, if $\|v\|_1 < p^k$ then we have an integer polynomial with the desired root.
Multivariate polynomial approach (MP)

- We call a vector \( \mathbf{v} \in L \) such that \( \|\mathbf{v}\|_1 < p^k \) a **target vector**.
- We need \( t \) least \( m \) algebraically independent target vectors.
- Elimination leads to \((r_1, \ldots, r_m)\).
- One then computes \( p = \gcd(N, a_1 - r_1) \).
- We call this process the **MP algorithm**.
- The case \((t, k) = (1, 1)\) gives the OL method, as noted by van Dijk et al.
  - Cohn-Heninger call \((t, k) = (1, 1)\) “unoptimised”.
- Does taking \( t > 1 \) gives rise to a better attack?
- When the number of ACD samples is large the best choice for MP algorithm is \((t, k) = (1, 1)\).
Multivariate polynomial approach (MP)

- Necessary condition for success using LLL is

\[ d \log_2(d) + d^2 \log_2(1.02) + d \rho \frac{mt}{m+1} + \gamma \binom{k+m}{m} \frac{k}{m+1} < k \eta d. \]

- This is equation (5.2) in our paper.

- Cohn-Heninger fix \( m \), set \( \beta = \frac{\eta}{\gamma} \ll 1 \), and impose \( t \approx \beta^{-1/m} k \), which means that \( t \gg k \).

- The lattice dimension in their method is

\[ \binom{t+m}{m} = O(t^m) = O(\beta^{-1} k^m) > \frac{\gamma}{\eta}. \]

This is the same dimension bound as previous methods (at least, when \( \rho \) is small).
Multivariate polynomial approach (MP)

- For large $m$, $\frac{mt}{m+1} \approx t$. To satisfy (5.2) need
  \[ t\rho < k \eta. \]

- Equation (5.2) implies, when $m$ is large,
  \[ d \rho t + \gamma \binom{k + m}{m} \frac{k}{m+1} < k \eta d. \]

- Dividing by $k$ and re-arranging gives
  \[ d > \frac{\gamma}{\eta - \frac{t}{k} \rho} \binom{k + m}{m} \frac{1}{m+1}. \]

Since $\frac{t}{k} \geq 1$ and $\binom{k+m}{m} \frac{1}{m+1} \geq 1$ we see that this is never better than the lattice dimension bound $d > \frac{\gamma}{\eta - \rho}$. 
Executive summary

- There is no theoretical reason why, when number of samples $m$ is large, the MP method should be better than the SDA or OL methods for any of the variants of the ACD problem.
- A special case $(t, k) = (1, 1)$ of the MP method gives the OL method. This was noted by van Dijk et al, and Cohn-Heninger call $(t, k) = (1, 1)$ “unoptimised”.
- Our practical experiments confirm this, and indeed show the MP algorithm with $(t, k) \neq (1, 1)$ is very slow due to solving systems of polynomial equations.
- When $m$ is very small then one can handle larger errors using the multivariate polynomial approach than SDA or OL (see ANTS 2012).
Pre-processing of the ACD samples

- Most important factor in the difficulty of the ACD problem is the ratio $\gamma/\eta$.
- If can lower $\gamma$ without changing the size of the errors then have an easier instance.
- Hence, we consider a pre-processing step where a large number of initial samples $x_i = pq_i + r_i$ are used to form new samples $x_j' = pq_j' + r_j'$ with $q_j'$ significantly smaller than $q_i$.
- Take differences $x_k - x_i$ for $x_k > x_i$ and $x_k \approx x_i$.
- Note that if $x_k \approx x_i$ then $q_k \approx q_i$ but $r_k$ and $r_i$ are not necessarily related at all.
- Hence $x_k - x_i = p(q_k - q_i) + (r_k - r_i)$ is an ACD sample for the same $p$, with smaller value for $q$ and a similar sized error $r$. 
Pre-processing of the ACD samples

- We also propose a **sample amplification** idea to convert a small list of samples into a large list, so that the method can be iterated.
- This approach looks stupid: Why not just build a lattice from all the samples.
- But the number of samples may be astronomically large.
Blum-Kalai-Wasserman (BKW) algorithm

- Our work is inspired by the BKW algorithm for learning parity with noise (LPN).
- In that case we have samples \((a, b)\) where \(a \in \mathbb{Z}_2^n\) is a vector of length \(n\) and \(b = a \cdot s + e\), where \(s \in \mathbb{Z}_2^n\) is a secret and \(e\) is a noise term which is usually zero.
- To obtain samples such that \(a = (1, 0, 0, \ldots, 0)\), or similar, iterate by adding samples \((a_k, b_k) + (a_i, b_i)\) where some coordinates of \(a_k\) and \(a_i\) agree.
- The result is an algorithm with subexponential complexity \(2^{n/\log(n)}\), compared with the naive algorithm (guessing all \(s \in \mathbb{Z}_2^n\)) which has complexity \(2^n\).
- In our context we do not have \((q_i, pq_i + r_i)\) but only \(x_i = pq_i + r_i\), however we can use the high-order bits of \(x_i\) as a proxy for the high order bits of \(q_i\) and hence perform a similar algorithm.
Preserving the sample size

- Fix a small bound $B = 2^b$ (e.g., $B = 16$) and select $B$ samples $x_1, \ldots, x_B$ such that the leading coefficients in base $B$ are all distinct.
- For each of the remaining $\tau - B$ samples, generate a new sample by subtracting the one with the same leading coefficient.
- The result is $\tau - B$ samples each of size $\gamma - b = \gamma - \log_2(B)$ bits.
- Easy to see this is stupid.
Aggressive shortening

- Sort the samples $x_1 \leq x_2 \leq \cdots \leq x_T$ and, for some small threshold $T = 2^{\gamma - \mu}$, generate new samples by subtracting $x_{i+1} - x_i$ when this difference is less than $T$.
- The new samples are of size at most $\gamma - \mu$ bits, but there are far fewer of them.
- The statistical distribution of such “ spacings” was considered by Pyke.
  It is shown that generic spacings have Exponential distributions.
- Eventually one has too few samples.
Sample amplification

- Generate new samples of about the same bitlength by taking sums/differences of the initial list of samples.
- Let $\mathcal{L} = \{x_1, \ldots, x_T\}$ be a list of ACD samples, with $x_k = pq_k + r_k$ having mean and variance given by
  
  $$\mu = \mathbb{E}(x_k) = p\mathbb{E}(q_k) = 2^{\gamma-1}$$
  
  and variance given by
  
  $$\text{Var}(x_k) = p^2\text{Var}(q_k) + \text{Var}(r_k) = \frac{1}{3}2^{2(\gamma-1)} + \frac{1}{12}2^{2\rho}$$
  
  $$= \frac{1}{3}2^{2(\gamma-1)} \left(1 + 2^{-2(\gamma-\rho)}\right).$$

- Generate $m$ random sums
  
  $$S_k = \sum_{i=1}^{\ell} x_{k,i} \quad [k = 1, \ldots, m],$$

  which have mean and variance given by
  
  $$\mathbb{E}(S_k) = l2^{\gamma-1}$$
  
  and
  
  $$\text{Var}(S_k) = \frac{1}{3}l2^{2(\gamma-1)} \left(1 + 2^{-2(\gamma-\rho)}\right).$$
Aggressive shortening

- Start with a list \( L = \{x_1, \ldots, x_\tau\} \) of ACD samples of mean value \( 2^{\gamma-1} \) and standard deviation \( \sigma_0 \approx 3^{-\frac{1}{2}} 2^{(\gamma-1)} \).
- Amplify this to a list of \( m \) samples \( S_k \).
- Sort the \( S_k \) to get the spacings \( S_{k+1} - S_k \).
- Store the \( \tau = m/2 \) “middle” spacings as input to the next iteration of the algorithm.
- After an appropriate number of iterations run the orthogonal lattice attack.
- Conclusion: It still doesn’t work, the number of iterations required is just too large.
Contributions

- We obtained a refined lower bound \((\gamma - \rho)/(\eta - \rho)\) on the dimension of lattices in the SDA and OL algorithms.
- We showed that all orthogonal lattice methods for ACD are basically the same.
- We showed the multivariate polynomial method is not better than other methods for cryptanalysis of homomorphic encryption schemes based on ACD.
- We explored an analogue of the BKW algorithm for ACD and showed that it doesn’t work.
Open problems

- Find improved algorithms for the CRT-ACD problem.
- Find improved algorithms for partial ACD (i.e., when one is given an exact multiple $pq_0$ of $p$).
Thank you for your attention