Random subgroups of braid groups: cryptanalysis of a braid group based cryptographic protocol

Vladimir Shpilrain (The City College of New York)

joint with

Alexei G. Myasnikov (McGill University)
Alexander Ushakov (Stevens Institute of Technology)

PKC 2006

Non-commutative cryptography

1. Adopting "commutative problems" (most notably, the discrete logarithm problem) in non-commutative situation. Example: Ko-Lee et. al. (CRYPTO 2000) use the conjugacy search problem: recover an $x \in G$ from given $g \in G$ and $h = g^x = x^{-1}gx$.

2. Using problems specific to non-commutative (semi)groups. Example: Anshel-Anshel-Goldfeld (Math. Res. Lett., 1999). Here the common secret key is of the form $xyx^{-1}y^{-1} = [x, y]$.

3. Using non-recursive decision problems, thus making cryptographic products secure against computationally unbounded adversary. (New mentality is required !)

Advantages of a non-commutative platform

1. Larger arsenal of (allegedly) computationally hard problems.
2. Introducing non-recursive decision problems to cryptography.
3. Larger key space, at a low cost.
4. Efficient multiplication (in some groups).

Weaknesses of a non-commutative platform

1. Factors are not "naturally" hidden in a product. Compare: $3 \cdot 7 = 21$, whereas $x \cdot y = xy$. Therefore, a "normal form" is required for hiding information.

2. "Marketing" disadvantage: security models are not well established, especially if the platform (semi)group is infinite. (New security model, based on the generic case complexity, is emerging.)

3. Insufficient accumulation of information on generic properties of elements, subgroups, etc.

What to do?

1. Combine commutative and non-commutative platforms to get "the best of both worlds", e. g. use matrices over something commutative. Example: Tillich-Zémor hash function (CRYPTO 1994)

2. Identify various types of attacks specific to non-commutative situation. Examples: "length-based" attacks (Hofheinz-Steinwandt, PKC 2003, and others), using one normal form to "unscramble" the other (Myasnikov-Shpilrain-Ushakov, CRYPTO 2005), etc.

In this talk:

A heuristic attack on the Anshel-Anshel-Goldfeld key exchange protocol (Math. Res. Lett. 1999, CT-RSA 2001):

- 99% success rate in recovering private keys
- 98% success rate in recovering shared keys.

Braid Group

B_n is a group of braids on n strands. It has a finite presentation by generators and relators:

$$B_n = \left\langle \begin{array}{ll} x_1, \ldots, x_{n-1}; & x_i x_j = x_j x_i \text{ if } |i-j| > 1, \\ & x_i x_{i+1} x_i = x_{i+1} x_i x_{i+1} \end{array} \right\rangle$$

A braid word w is a finite sequence

$$w = x_{i_1}^{\varepsilon_1} \ldots x_{i_k}^{\varepsilon_k},$$

where $1 \leq i_j \leq n-1$ and $\varepsilon_j = \pm 1$.

Each element of B_n can be represented by a braid word. Hence, we work with elements of B_n as with braid words.

The Anshel-Anshel-Goldfeld protocol (a.k.a. Arithmetica key exchange)

- B_n – fixed braid group
- $k, m \in \mathbb{Z}$ – fixed parameters.

---

1) Alice chooses randomly:

   - braid words $\{a_1, \ldots, a_k\}$ – generators of Alice's public subgroup.
   - a product $A = a_{i_1}^{\varepsilon_1} \ldots a_{i_m}^{\varepsilon_m}$ – Alice's private key.

2) Bob chooses randomly:

   - braid words $\{b_1, \ldots, b_k\}$ – generators of Bob's public subgroup.
   - a product $B = b_{j_1}^{\delta_1} \ldots b_{j_m}^{\delta_m}$ – Bob's private key.

Arithmetica key exchange
The protocol

1) Alice sends normal forms $\hat{b}_i = N(A^{-1}b_iA)$ $(i = 1, \ldots, k)$ to Bob.

2) Bob sends normal forms $\hat{a}_i = N(B^{-1}a_iB)$ $(i = 1, \ldots, k)$ to Alice.

3) Alice computes $K_A = A^{-1} \cdot \hat{a}_{i_1}^{\varepsilon_1} \cdot \ldots \cdot \hat{a}_{i_m}^{\varepsilon_m}$.

4) Bob computes $K_B = \left[\hat{b}_{j_1}^{\delta_1} \cdot \ldots \cdot \hat{b}_{j_m}^{\delta_m}\right]^{-1} \cdot B^{-1}$.

Then $K_A = K_B = A^{-1}B^{-1}AB$ in B_n.

Security of the protocol

Relies on the computational difficulty of the Multiple Conjugacy Search Problem:

- Given $(a_1, \ldots, a_k)$, $(\hat{a}_1, \ldots, \hat{a}_k) \in B_n^k$, and $(b_1, \ldots, b_k) \in B_n^k$:
- Find an element $B \in \langle b_1, \ldots, b_k \rangle$ such that $B^{-1} a_i B = \hat{a}_i$ (provided that at least one such B exists).

Arithmetica key exchange (parameters)

Initially (1999):

- Braid group - B_{80}
- $k = 20$; $m = 100$
- $5 \leq |a_i| \leq 8$.

Later (2001):

- Braid group - B_{150}
- $k = 20$; $m = 100$
- $13 \leq |a_i| \leq 15$.

Subgroup attack

Idea. Transform the pair $(a_1, \ldots, a_k)$, $(\hat{a}_1, \ldots, \hat{a}_k)$ of conjugate tuples to another pair $(c_1, \ldots, c_K)$, $(\hat{c}_1, \ldots, \hat{c}_K)$ of conjugate tuples, such that:

- For all $X \in B_n$

  $$X^{-1} a_i X = \hat{a}_i \ (i = 1, \ldots, k) \iff X^{-1} c_j X = \hat{c}_j \ (j = 1, \ldots, k).$$

- $(c_1, \ldots, c_K)$ is "simpler" than $(a_1, \ldots, a_k)$ (its elements are shorter).

Experiments

1) Generated 100 random pairs of conjugate tuples of braid words

$$(a_1, \ldots, a_k), (\hat{a}_1, \ldots, \hat{a}_k)$$

($k = 20$, $a_i \in B_{80}$, $5 \leq |a_i| \leq 8$);
2) For each pair used a sequence of transformations to obtain pairs

$$(c_1, \ldots, c_K), (\hat{c}_1, \ldots, \hat{c}_K)$$

of conjugate tuples as above, such that $(c_1, \ldots, c_K)$ is:

- $(x_1, \ldots, x_{79})$ in 63 cases
- $(x_1, \ldots, x_{i-1}, x_i^2, x_{i+1}, \ldots, x_{79})$ in 25 cases
- $(x_1, \ldots, x_{i-1}, x_i^2, x_{i+1}, \ldots, x_{j-1}, x_j^2, x_{j+1}, \ldots, x_{79})$ in 5 cases
- $(x_1, \ldots, x_{i-1}, x_i^2, x_i x_{i+1}^2 x_i, x_{i+2}, \ldots, x_{79})$ in 5 cases
- $(x_1, \ldots, x_{i-1}, x_i^2, x_i x_{i+1}^2 x_i, x_{i+1}, \ldots, x_{j-1}, x_j^3, x_{j+1}, \ldots, x_{79})$ in 1 case
- $(x_1, \ldots, x_{i-1}, x_i^{-1} x_{i+1} x_i, x_{i+2} \ldots, x_{79})$ in 1 case

Experimental results

Therefore, we obtained equivalent pairs of tuples which:

- in 99% cases consists of short positive words
- in 100% the summit set of $(c_1, \dots, c_K)$ is small
- in 100% the pointwise centralizer of $(c_1, \dots, c_K)$ coincides with the center of B_n (generated by Δ^2).

Impact on the security of the Arithmetica key exchange

We have a simplified pair of tuples: $(c_1, \ldots, c_K)$, $(\hat{c}_1, \ldots, \hat{c}_K)$

- apply the cycling technique described in [Lee, Lee] for $(\hat{c}_1, \ldots, \hat{c}_K)$ to obtain a tuple

$$(\hat{c}'_1, \ldots, \hat{c}'_K)$$

  conjugated to $(\hat{c}_1, \ldots, \hat{c}_K)$ (with actual conjugator) which belongs to the summit set of $(c_1, \ldots, c_K)$;

- using technique described in [Gonzalez-Meneses] construct the summit set of $(c_1, \ldots, c_K)$ and solve the conjugacy problem for

$$(\hat{c}'_1, \ldots, \hat{c}'_K) \text{ and } (c_1, \ldots, c_K);$$

- combine the obtained conjugators and denote the result by X.

$X = B \cdot \Delta^{2s}$ since in all cases the centralizer of $(c_1, \ldots, c_K)$ is $\langle \Delta^2 \rangle$. Therefore, X is "as good as" Bob's private key B.

Would increasing the parameters save Arithmetica key exchange?

Probably not, but there is no proof at this time.

Conjecture: Asymptotically, k-generated subgroups of B_n are free (n, k - fixed, $l \to \infty$).

The only hope for Arithmetica key exchange: Find a threshold function $F(n,k,l) = 0$ such that: if $F(n,k,l) < 0$, then most of k-tuples of braid words of length l generate B_n, and if $F(n,k,l) > 0$, then most of k-tuples of braid words of length l do not generate B_n (as $n \to \infty$).