

# SUBGROUP ATTACK

**Sasha Ushakov**

Stevens Institute of Technology

aushakov@stevens.edu

(joint work w. A. G. Miasnikov and V. Shpilrain)

## Main results

A heuristic attack on **Anshel-Anshel-Goldfeld key exchange protocol**:

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

(for original and newly updated parameter values).

# A braid

... a braid on  $n$  strands is obtained by laying down  $n$  parallel strings and intertwining them without losing track of direction (no knots) ...



Figure 1: A braid on 4 strands.

- 
- colors of strands do not have any mathematical meaning
  - strands are numbered from 1 to  $n$
  - direction is from left to right

## Equivalence on braids

Two braids are equivalent if there is an isotopy between them (isotopy corresponds physically to a motion of the strands, with the endpoints being kept fixed).

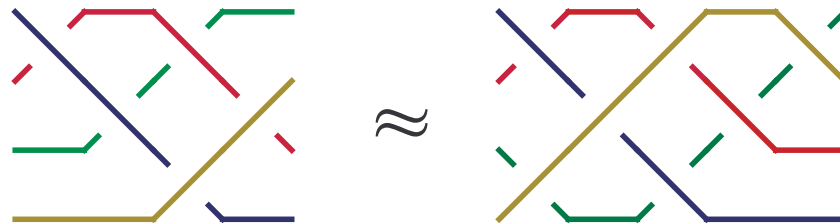


Figure 2: Equivalent braids.

# Trivial braid



Figure 3: Trivial braid.

# Multiplication of braids

Product = concatenation

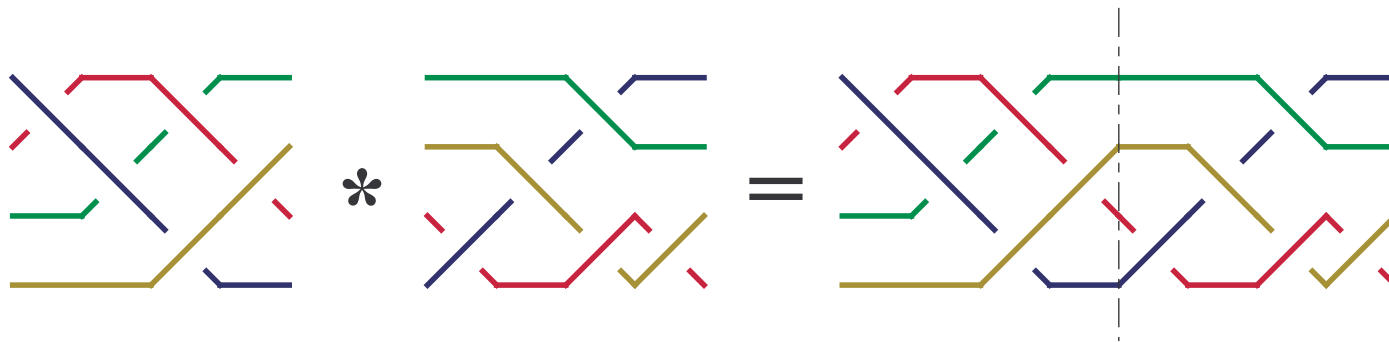


Figure 4: Product of two braids on 4 strands.

# Inverting braids

Inversion = taking mirror image in a vertical plane



Figure 5:

## Generators for braids

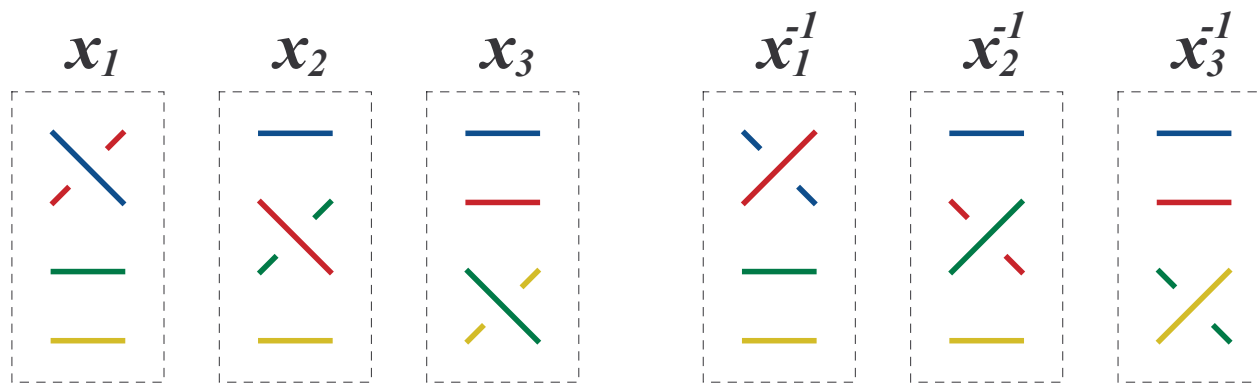


Figure 6: Generators and their inverses for braids on 4 strands.

# Braid relations

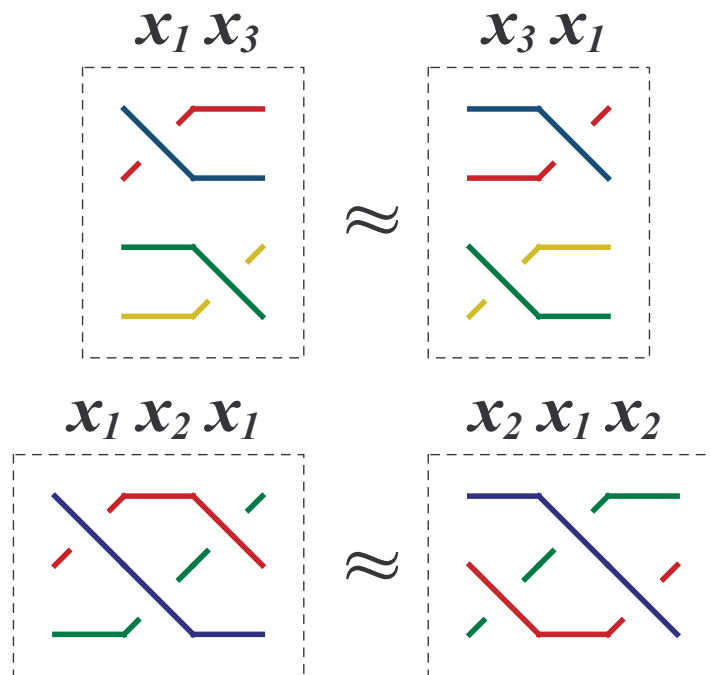


Figure 7: Relations in braids.

# Braid Group

---

$B_n$  is a group of braids on  $n$  strands. It has finite combinatorial presentation:

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

---

A **braid word**  $w$  is a finite sequence

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

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.

## Dehornoy handle-free form

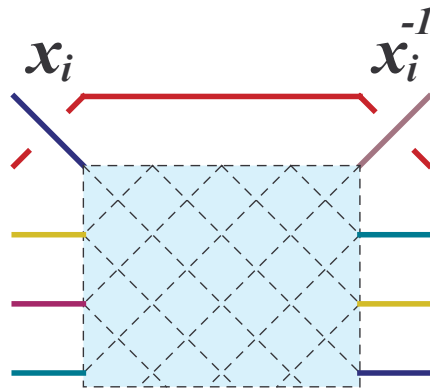


Figure 8: Handle.

$D(u)$  - **Dehornoy handle-free form** of a braid word  $u$ .

- $u =_{B_n} 1$  if and only if  $D(u) = \varepsilon$ .
- $D$  has a property to reduce lengths of "generic" braid words. Therefore, to shorten  $u$  one can try to compute  $D(u)$ .
- $D(D(u)) = D(u)$ .

# Commutator key exchange (preliminaries)

- $G = \langle X; R \rangle$  – fixed group;
  - $k, m \in \mathbb{Z}$  – fixed parameters.
- 

1) Alice chooses random:

- elements  $\{a_1, \dots, a_k\}$  – Alice's public subgroup.
- a product  $A = a_{i_1}^{\varepsilon_1} \dots a_{i_m}^{\varepsilon_m}$  – Alice's private key.

2) Bob chooses random:

- elements  $\{b_1, \dots, b_k\}$  – Bob's public subgroup.
- a product  $B = b_{j_1}^{\delta_1} \dots b_{j_m}^{\delta_m}$  – Bob's private key.

## Commutator key exchange (protocol)

- 1) Alice sends to Bob braids  $\hat{b}_i = D(A^{-1}b_iA)$  ( $i = 1, \dots, k$ ).
- 2) Bob sends to Alice braids  $\hat{a}_i = D(B^{-1}a_iB)$  ( $i = 1, \dots, k$ ).
- 3) Alice computes  $K_A = A^{-1} \cdot \hat{a}_{i_1}^{\varepsilon_1} \cdot \dots \cdot \hat{a}_{i_m}^{\varepsilon_m}$ .
- 4) Bob computes  $K_B = \left[ \hat{b}_{j_1}^{\delta_1} \cdot \dots \cdot \hat{b}_{j_m}^{\delta_m} \right]^{-1} \cdot B$ .

Clearly,  $K_A =_{B_n} K_B =_{B_n} A^{-1}B^{-1}AB$ .

# AAG key exchange (parameters)

---

## Initially:

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

---

## Recently:

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

## Security of the protocol

Relies on the computational difficulty of the **Subgroup-Related Simultaneous Conjugacy Search Problem**:

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

**Remark.** Since the shared key  $K$  is a commutator  $[A, B]$  it follows that to recover  $K$  it is enough to find  $B$  (and  $A$ ) up to a central element of  $G$ .

# Subgroup attack

For a general commutator key exchange ...

**Idea.** Transform the conjugated tuples  $(a_1, \dots, a_k)$ ,  $(\hat{a}_1, \dots, \hat{a}_k)$  into conjugated tuples  $(c_1, \dots, c_K)$ ,  $(\hat{c}_1, \dots, \hat{c}_K)$  satisfying properties:

- For all  $X \in B_n$

$$X^{-1}a_iX = \hat{a}_i \quad (i = 1, \dots, k) \iff X^{-1}c_jX = \hat{c}_j \quad (j = 1, \dots, k).$$

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

## Subgroup attack (transformations - I)

Let  $1 \leq i, j \leq k$ ,  $i \neq j$ , and  $\varepsilon = \pm 1$ . Nielsen transformation of conjugated tuples is:

$$\left\{ \begin{array}{l} (a_1, \dots, a_i, \dots, a_j, \dots, a_k) \rightarrow (a_1, \dots, a_i, \dots, a_j a_i^\varepsilon, \dots, a_k) \\ (\hat{a}_1, \dots, \hat{a}_i, \dots, \hat{a}_j, \dots, \hat{a}_k) \rightarrow (\hat{a}_1, \dots, \hat{a}_i, \dots, \hat{a}_j \hat{a}_i^\varepsilon, \dots, \hat{a}_k) \end{array} \right.$$

or

$$\left\{ \begin{array}{l} (a_1, \dots, a_i, \dots, a_j, \dots, a_k) \rightarrow (a_1, \dots, a_i, \dots, a_i^\varepsilon a_j, \dots, a_k) \\ (\hat{a}_1, \dots, \hat{a}_i, \dots, \hat{a}_j, \dots, \hat{a}_k) \rightarrow (\hat{a}_1, \dots, \hat{a}_i, \dots, \hat{a}_i^\varepsilon \hat{a}_j, \dots, \hat{a}_k) \end{array} \right.$$

## Subgroup attack (transformations - II)

Let  $a = a_{i_1} \dots a_{i_m}$  and  $\hat{a} = \hat{a}_{i_1} \dots \hat{a}_{i_m}$ . Then **extension** of conjugated tuples is:

$$\left\{ \begin{array}{l} (a_1, \dots, a_i, \dots, a_j, \dots, a_k) \rightarrow (a_1, \dots, a_i, \dots, a_j, \dots, a_k, a) \\ (\hat{a}_1, \dots, \hat{a}_i, \dots, \hat{a}_j, \dots, \hat{a}_k) \rightarrow (\hat{a}_1, \dots, \hat{a}_i, \dots, \hat{a}_j, \dots, \hat{a}_k, \hat{a}) \end{array} \right.$$

**Removal** of a trivial element  $a_i$ :

$$\left\{ \begin{array}{l} (a_1, \dots, a_{i-1}, \mathbf{1}, a_{i+1}, \dots, a_k) \rightarrow (a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_k) \\ (\hat{a}_1, \dots, \hat{a}_{i-1}, \mathbf{1}, \hat{a}_{i+1}, \dots, \hat{a}_k) \rightarrow (\hat{a}_1, \dots, \hat{a}_{i-1}, \hat{a}_{i+1}, \dots, \hat{a}_k) \end{array} \right.$$

## Algorithm (for AAG)

Given: conjugated tuples  $(a_1, \dots, a_k)$  and  $(\hat{a}_1, \dots, \hat{a}_k)$ :

---

- If there is a Nielsen transformation which reduces the total length of  $(a_1, \dots, a_k)$  then apply it.
- If for some pair of braids  $a_i, a_j$  and  $\varepsilon = \pm 1, \delta = \pm 1$  the product

$$a' = a_i^{2\varepsilon} a_j^\delta a_i^{-\varepsilon} a_j^{-\delta} a_i^{-\varepsilon}$$

has length 2 then extend tuples with  $a'$  and

$$\hat{a}' = \hat{a}_i^{2\varepsilon} \hat{a}_j^\delta \hat{a}_i^{-\varepsilon} \hat{a}_j^{-\delta} \hat{a}_i^{-\varepsilon}$$

- If  $|a_i| = |a_j| = 2, |a_i^\varepsilon a_j^\delta| = 2$ , and both  $(a_i^\varepsilon a_j^\delta)^{\pm 1}$  do not belong to  $(a_1, \dots, a_k)$  then extend tuples with  $a_i^\varepsilon a_j^\delta$  and  $\hat{a}_i^\varepsilon \hat{a}_j^\delta$ .
- 

Proceed while transformations are applicable.

Procedure stops in finite time on any input.

# Experiments

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

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

( $k = 20, a_i \in B_{80}, 5 \leq |a_i| \leq 8$ );

2) for each pair use the procedure above and denote the result by

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

**Results:**  $(c_1, \dots, c_K)$  is

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

# Experiments

Therefore, we obtained equivalent pairs of tuples which:

- in 99% cases consists of short positive words;
- in 100% 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  $\Delta^2$ ).

# Impact on the security of AAG key exchange

We have a simplified pair  $(c_1, \dots, c_K), (\hat{c}_1, \dots, \hat{c}_K)$

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

$$(\tilde{c}'_1, \dots, \tilde{c}'_K)$$

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

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

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

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

$X = B\Delta^{2s}$  since in all cases centralizer of  $(c_1, \dots, c_K)$  is  $\langle \Delta^2 \rangle$ .

Therefore,  $X$  can play the role of Bob's key  $B$ .

## Why does it work?

Consider two particular braid words:

$$w_1 = x_{71}x_{47}x_{11}x_{45}^{-1}x_9x_6x_{72}^{-1}$$

and

$$w_2 = x_{64}x_{32}^{-1}x_{39}^{-1}x_{17}x_8x_{26}x_{31}^{-1}x_{78}.$$

Check that  $w_1^2w_2w_1^{-1}w_2^{-1}w_1^{-1} = x_9^2x_8x_9^{-1}x_8^{-1}x_9^{-1} = x_9x_8^{-1}$ .

---

Generate 2 random braid words  $w_1, w_2$  from  $B_{80}$  of lengths in the interval  $[5, 8]$ :

- it is "very likely" that they will commute;
- if they do not commute then it is "very likely" that it happens only because  $w_1$  involves  $x_i^\varepsilon$  and  $w_2$  involves  $x_{i+1}^\delta$  ( $\varepsilon, \delta = \pm 1$ ). In this case a braid word  $x_i^\varepsilon x_{i+1}^{-\varepsilon}$  can be deduced from  $w_1$  and  $w_2$ .

There are three principal stages of the execution flow:

- 1) First, the algorithm extends tuples with braid words of the type  $x_i x_{i+1}^{-1}$  using the pattern  $w_1^2 w_2 w_1^{-1} w_2^{-1} w_1^{-1}$ ;
- 2) Second, it extends tuples with braid words  $x_i^\varepsilon x_j^\delta$  using pattern  $w_i^\varepsilon w_j^\delta$ .
- 3) Finally, (when some critical number of braid words of the type  $x_i^\varepsilon x_j^\delta$  is generated) the algorithm reduces one of the  $w_j$  to  $x_i$  (using Nielsen transformations) and all other  $w_j$ 's reduce to generators too.

## What if we increase the lengths of $w_1$ and $w_2$ ?

If  $|w_1|, |w_2| > 8$  then it is likely that more pairs of letters in  $w_1$  and  $w_2$  will not commute. If two pairs do not commute then it is likely that:

$$w_1^2 w_2 w_1^{-1} w_2^{-1} w_1^{-1} = (x_i x_{i+1}^{-1})(x_j x_{j+1}^{-1}).$$

Hence, if such correlation is expected then one can improve the performance of the algorithm by allowing extension of the tuples with words of length 4 (at a cost of reducing the speed).

---

Our experiments show that random tuples  $(a_1, \dots, a_k)$ , where  $k = 20$ ,  $a_i \in B_{80}$ , and  $13 \leq |a_i| \leq 15$  very often generate the whole group  $B_{80}$ .

---

When the lengths are increased to  $19 \leq |a_i| \leq 21$  our algorithm fails to simplify tuples.

---

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

---

**Problem:** 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  $n \rightarrow \infty$ ).