Knot-based Key Exchange Protocol

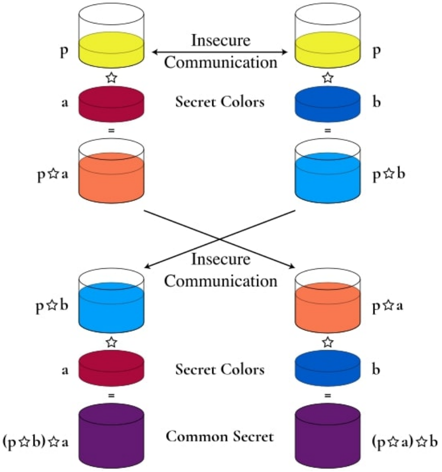
Silvia Sconza,
joint work with Arno Wildi

CrypTO Seminars, Politecnico di Torino

March 22nd, 2024

Table of Contents

Diffie-Hellman Key Exchange



[Picture from Borradaile, G. "Defend Dissent." Corvallis: Oregon State University, 2021.]

Diffie-Hellman Key Exchange

Diffie-Hellman Key Exchange (DHKE), 1976 [2]

1. Alice and Bob publicly agree on a cyclic finite group G and a generator g.

Diffie-Hellman Key Exchange (DHKE), 1976 [2]

1. Alice and Bob publicly agree on a cyclic finite group G and a generator g.
2. Alice chooses $a \in \{1, \ldots, \mathrm{ord}(G)\}$, computes g^a and sends it to Bob. Her secret key is a.

Diffie-Hellman Key Exchange

Diffie-Hellman Key Exchange (DHKE), 1976 [2]

1. Alice and Bob publicly agree on a cyclic finite group G and a generator g.
2. Alice chooses $a \in \{1, \ldots, \mathrm{ord}(G)\}$, computes g^a and sends it to Bob. Her secret key is a.
3. Bob chooses $b \in \{1, \ldots, \mathrm{ord}(G)\}$, computes g^b and sends it to Alice. His secret key is b.

Diffie-Hellman Key Exchange

Diffie-Hellman Key Exchange (DHKE), 1976 [2]

1. Alice and Bob publicly agree on a cyclic finite group G and a generator g.
2. Alice chooses $a \in \{1, \dots, \mathrm{ord}(G)\}$, computes g^a and sends it to Bob. Her secret key is a.
3. Bob chooses $b \in \{1, \dots, \mathrm{ord}(G)\}$, computes g^b and sends it to Alice. His secret key is b.
4. Alice computes $(g^b)^a = g^{ba}$.

Diffie-Hellman Key Exchange (DHKE), 1976 [2]

1. Alice and Bob publicly agree on a cyclic finite group G and a generator g.
2. Alice chooses $a \in \{1, \ldots, \mathrm{ord}(G)\}$, computes g^a and sends it to Bob. Her secret key is a.
3. Bob chooses $b \in \{1, \ldots, \mathrm{ord}(G)\}$, computes g^b and sends it to Alice. His secret key is b.
4. Alice computes $(g^b)^a = g^{ba}$.
5. Bob computes $(g^a)^b = g^{ab}$.

Diffie-Hellman Key Exchange

Diffie-Hellman Key Exchange (DHKE), 1976 [2]

1. Alice and Bob publicly agree on a cyclic finite group G and a generator g.
2. Alice chooses $a \in \{1, \ldots, \mathrm{ord}(G)\}$, computes g^a and sends it to Bob. Her secret key is a.
3. Bob chooses $b \in \{1, \ldots, \mathrm{ord}(G)\}$, computes g^b and sends it to Alice. His secret key is b.
4. Alice computes $(g^b)^a = g^{ba}$.
5. Bob computes $(g^a)^b = g^{ab}$.

The secret common key is $g^{ba} = g^{ab}$.

Diffie-Hellman Key Exchange

Diffie-Hellman Key Exchange (DHKE), 1976 [2]

1. Alice and Bob publicly agree on a cyclic finite group G and a generator g.
2. Alice chooses $a \in \{1, \ldots, \mathrm{ord}(G)\}$, computes g^a and sends it to Bob. Her secret key is a.
3. Bob chooses $b \in \{1, \ldots, \mathrm{ord}(G)\}$, computes g^b and sends it to Alice. His secret key is b.
4. Alice computes $(g^b)^a = g^{ba}$.
5. Bob computes $(g^a)^b = g^{ab}$.

The secret common key is $g^{ba} = g^{ab}$.

- Diffie-Hellman Problem (DHP): Let G be a finite cyclic group and let g be a generator. Given g^a and g^b, find g^{ab}.

Group actions

Given G an abelian group with identity element e and a set X, a group action of G on X is a map

$$\begin{aligned} \star \colon G \times X &\longrightarrow X \\ (g, x) &\mapsto g \star x \end{aligned}$$

s.t. $e \star x = x$ and $g \star (h \star x) = (gh) \star x$ for all $g, h \in G$ and $x \in X$.

Given G an abelian group with identity element e and a set X, a group action of G on X is a map

$$\begin{aligned} \star \colon G \times X &\longrightarrow X \\ (g, x) &\mapsto g \star x \end{aligned}$$

s.t. $e \star x = x$ and $g \star (h \star x) = (gh) \star x$ for all $g, h \in G$ and $x \in X$.

Example:

Group actions

Given G an abelian group with identity element e and a set X, a group action of G on X is a map

$$\begin{aligned} \star \colon G \times X &\longrightarrow X \\ (g, x) &\mapsto g \star x \end{aligned}$$

s.t. $e \star x = x$ and $g \star (h \star x) = (gh) \star x$ for all $g, h \in G$ and $x \in X$.

Example: Let X be a cyclic finite group of order p

Group actions

Given G an abelian group with identity element e and a set X, a group action of G on X is a map

$$\begin{aligned} \star \colon G \times X &\longrightarrow X \\ (g, x) &\mapsto g \star x \end{aligned}$$

s.t. $e \star x = x$ and $g \star (h \star x) = (gh) \star x$ for all $g, h \in G$ and $x \in X$.

Example: Let X be a cyclic finite group of order p and $G = \mathbb{Z}_p^\times$.

Group actions

Given G an abelian group with identity element e and a set X, a group action of G on X is a map

$$\begin{aligned} \star\colon G \times X &\longrightarrow X \\ (g, x) &\mapsto g \star x \end{aligned}$$

s.t. $e \star x = x$ and $g \star (h \star x) = (gh) \star x$ for all $g, h \in G$ and $x \in X$.

Example: Let X be a cyclic finite group of order p and $G = \mathbb{Z}_p^\times$. Then

$$\begin{aligned} \mathbb{Z}_p^\times \times X &\longrightarrow X \\ (n, x) &\mapsto x^n \end{aligned}$$

is an **action** of $\mathbb{Z}_p^\times$ over X.

Generalised DHKE

Generalised Diffie-Hellman Key Exchange

1. Alice and Bob publicly agree on an abelian group G, an action $\star$ of G on a finite set X and an element $x \in X$.
2. Alice chooses $a \in G$, computes $a \star x$ and sends it to Bob. Her secret key is a.
3. Bob chooses $b \in G$, computes $b \star x$ and sends it to Alice. His secret key is b.
4. Alice computes $a \star (b \star x)$.
5. Bob computes $b \star (a \star x)$.

The secret common key is $(ab) \star x = (ba) \star x$.

Generalised DHKE

Generalised Diffie-Hellman Key Exchange

1. Alice and Bob publicly agree on an abelian group G, an action $\star$ of G on a finite set X and an element $x \in X$.
2. Alice chooses $a \in G$, computes $a \star x$ and sends it to Bob. Her secret key is a.
3. Bob chooses $b \in G$, computes $b \star x$ and sends it to Alice. His secret key is b.
4. Alice computes $a \star (b \star x)$.
5. Bob computes $b \star (a \star x)$.

The secret common key is $(ab) \star x = (ba) \star x$.

- Diffie-Hellman Group Action Problem (DHGAP): Let G, X and $\star$ as above. Given $x, y, z \in X$ such that $y = g \star x$ and $z = h \star x$ for some $g, h \in G$, find $(gh) \star x$.

Semigroups and semigroup actions

A semigroup is a set S together with a *binary operation* $\cdot : S \times S \to S$ that satisfies the associative property.

A semigroup is a set S together with a *binary operation* $\cdot : S \times S \to S$ that satisfies the associative property.

Given S an abelian semigroup and a set X, an S-action on X (or a semigroup action of S on X) is a map

$$\begin{aligned} \star : S \times X &\longrightarrow X \\ (s, x) &\mapsto s \star x \end{aligned}$$

s.t. $s \star (r \star x) = (s \cdot r) \star x$ for all $s, r \in S$ and $x \in X$

Generalised DHKE

Generalised Diffie-Hellman Key Exchange [4]

1. Alice and Bob publicly agree on an abelian semigroup S, an S-action $\star$ on a finite set X and an element $x \in X$.

Generalised DHKE

Generalised Diffie-Hellman Key Exchange [4]

1. Alice and Bob publicly agree on an abelian semigroup S, an S-action $\star$ on a finite set X and an element $x \in X$.
2. Alice chooses $a \in S$, computes $a \star x$ and sends it to Bob. Her secret key is a.
3. Bob chooses $b \in S$, computes $b \star x$ and sends it to Alice. His secret key is b.
4. Alice computes $a \star (b \star x)$.
5. Bob computes $b \star (a \star x)$.

The secret common key is $(ab) \star x = (ba) \star x$.

Generalised Diffie-Hellman Key Exchange [4]

1. Alice and Bob publicly agree on an abelian semigroup S, an S-action $\star$ on a finite set X and an element $x \in X$.
2. Alice chooses $a \in S$, computes $a \star x$ and sends it to Bob. Her secret key is a.
3. Bob chooses $b \in S$, computes $b \star x$ and sends it to Alice. His secret key is b.
4. Alice computes $a \star (b \star x)$.
5. Bob computes $b \star (a \star x)$.

The secret common key is $(ab) \star x = (ba) \star x$.

• Diffie-Hellman Semigroup Action Problem (DHSAP): Let S, X and $\star$ as above. Given $x, y, z \in X$ such that $y = s \star x$ and $z = r \star x$ for some $s, r \in S$, find $(gh) \star x$.

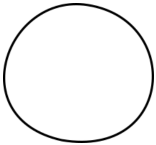
Table of Contents

Definitions

A *knot* is a smooth embedding $\mathbb{S}^1 \to \mathbb{R}^3$, considered up to ambient isotopy.

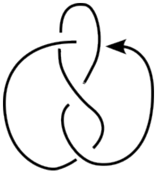
Definitions

A *knot* is a smooth embedding $\mathbb{S}^1 \to \mathbb{R}^3$, considered up to ambient isotopy.



Unknot $\mathcal{U}$

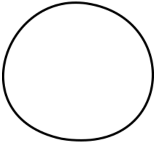
Trefoil knot



Oriented Figure-Eight knot

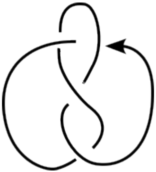
Definitions

A *knot* is a smooth embedding $\mathbb{S}^1 \to \mathbb{R}^3$, considered up to ambient isotopy.



Unknot $\mathcal{U}$

Trefoil knot



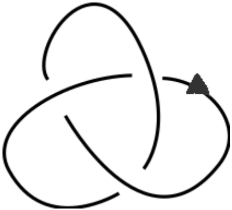
Oriented Figure-Eight knot

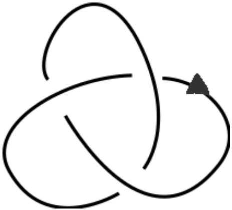
N.B.: We will consider just oriented knots.

Connected sum

Given two oriented knots K and K', we can define the *connected sum* $K \# K'$: cut the two knots and glue the corresponding ends (given by the orientation).

Example:





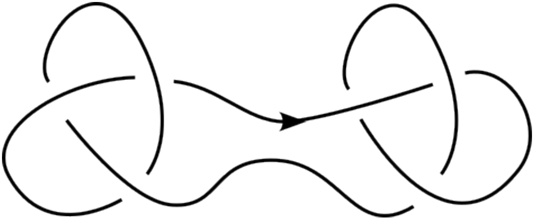
Given two oriented knots K and K', we can define the *connected sum* $K\#K'$: cut the two knots and glue the corresponding ends (given by the orientation).

Example:

Definitions

Given two oriented knots K and K', we can define the *connected sum* $K\#K'$: cut the two knots and glue the corresponding ends (given by the orientation).

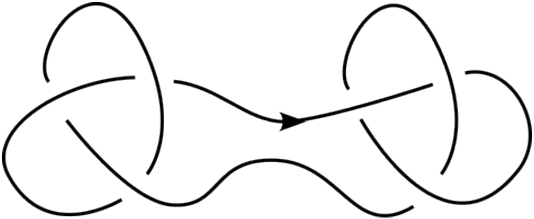
Example:



Definitions

Given two oriented knots K and K', we can define the *connected sum* $K \# K'$: cut the two knots and glue the corresponding ends (given by the orientation).

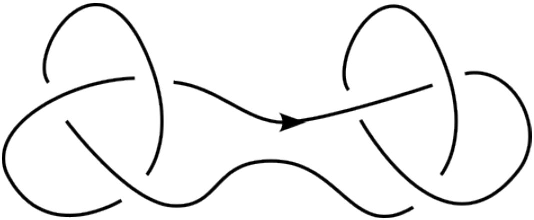
Example:



N.B.: With this operation, the set of oriented knots forms an abelian semigroup: $(\mathbf{oKnots}, \#, \mathcal{U})$.

Given two oriented knots K and K', we can define the *connected sum* $K \# K'$: cut the two knots and glue the corresponding ends (given by the orientation).

Example:

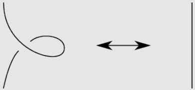


- Decomposition Problem: Given a knot K, find its prime decomposition $K = K_1 \# \cdots \# K_n$.

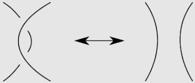
Definitions

Theorem (Reidemeister):

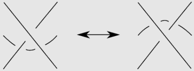
Two knots are the same if and only if they are related by a finite sequence of the Reidemeister moves:



R1



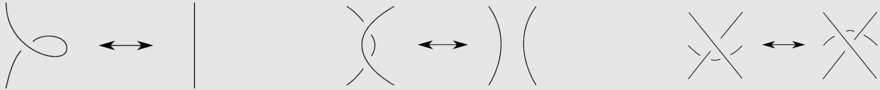
R2



R3

Theorem (Reidemeister):

Two knots are the same if and only if they are related by a finite sequence of the Reidemeister moves:



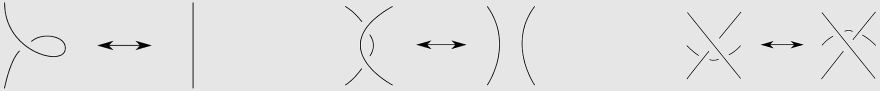
R1 R2 R3

- Recognition Problem: Given two knot diagrams K and K'. Do they represent the same knot?

Definitions

Theorem (Reidemeister):

Two knots are the same if and only if they are related by a finite sequence of the Reidemeister moves:



R1 R2 R3

• Recognition Problem: Given two knot diagrams K and K'. Do they represent the same knot?

↑ This is a hard mathematical problem. ↑

Invariants

To classify knots, one studies knot invariants, which are functions that do not change under Reidemeister moves.

Invariants

To classify knots, one studies knot invariants, which are functions that do not change under Reidemeister moves.

Fact: All known computable invariants are not complete.

Invariants

To classify knots, one studies knot invariants, which are functions that do not change under Reidemeister moves.

Fact: All known computable invariants are not complete.

We will use finite type invariants [3].

Invariants

To classify knots, one studies knot invariants, which are functions that do not change under Reidemeister moves.

Fact: All known computable invariants are not complete.

We will use finite type invariants [3].

Conjecture: The set of all finite type invariants distinguish knots.

Invariants

To classify knots, one studies knot invariants, which are functions that do not change under Reidemeister moves.

Fact: All known computable invariants are not complete.

We will use finite type invariants [3].

Conjecture: The set of all finite type invariants distinguish knots.

Fact: A finite type invariant of type d can be computed in

$$\mathcal{O}(c^d),$$

where c is the number of crossings of the knot.

Fixed a $d \in \mathbb{N}$, we can choose between <u>several distinct</u> finite type invariants of type d.

d	0	1	2	3	4	5	6
# d-Finite type invariants	1	1	2	3	6	10	19

d	7	8	9	10	11	12
# d-Finite type invariants	33	60	104	184	316	548

Encoding knots

Consider a planar representation of a knot K.

Encoding knots

Consider a planar representation of a knot K.

- Choose a starting point and an orientation. Enumerate the edges starting from 1, following the orientation.

Encoding knots

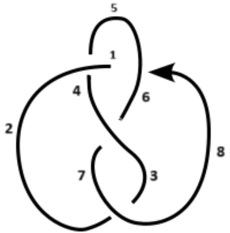
Consider a planar representation of a knot K.

- Choose a starting point and an orientation. Enumerate the edges starting from 1, following the orientation.
- To each crossing, we associate a list of four edges:
  (i) starting from the incoming undergoing edge;
  (ii) ordering the edges counterclockwise.

Encoding knots

Consider a planar representation of a knot K.

- Choose a starting point and an orientation. Enumerate the edges starting from 1, following the orientation.
- To each crossing, we associate a list of four edges:
  (i) starting from the incoming undergoing edge;
  (ii) ordering the edges counterclockwise.



[X[4,1,5,2], X[2,8,3,7], X[6,4,7,3], X[8,5,1,6]]

Table of Contents

First idea

Knot-based Key Exchange I

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.

First idea

Knot-based Key Exchange I

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A \# K$ and sends it to Bob. Her secret key is A.

First idea

Knot-based Key Exchange I

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A \# K$ and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B \# K$ and sends it to Alice. His secret key is B.

First idea

Knot-based Key Exchange I

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$ and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$ and sends it to Alice. His secret key is B.
4. Alice computes $A\#(B\#K) = A\#B\#K$.

First idea

Knot-based Key Exchange I

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A \# K$ and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B \# K$ and sends it to Alice. His secret key is B.
4. Alice computes $A \#(B \# K) = A \# B \# K$.
5. Bob computes $B \#(A \# K) = B \# A \# K$.

First idea

Knot-based Key Exchange I

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A \# K$ and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B \# K$ and sends it to Alice. His secret key is B.
4. Alice computes $A \#(B \# K) = A \# B \# K$.
5. Bob computes $B \#(A \# K) = B \# A \# K$.

The secret common key is $A \# B \# K = B \# A \# K$.

First idea

Knot-based Key Exchange I

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$ and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$ and sends it to Alice. His secret key is B.
4. Alice computes $A\#(B\#K) = A\#B\#K$.
5. Bob computes $B\#(A\#K) = B\#A\#K$.

The secret common key is $A\#B\#K = B\#A\#K$.

Problem I: In this case, given $A\#K$ and K, it is easy to find A.

Knot-based Key Exchange I

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$ and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$ and sends it to Alice. His secret key is B.
4. Alice computes $A\#(B\#K) = A\#B\#K$.
5. Bob computes $B\#(A\#K) = B\#A\#K$.

The secret common key is $A\#B\#K = B\#A\#K$.

Problem I: In this case, given $A\#K$ and K, it is easy to find A.

We need to "complicate" $A\#K$ and $B\#K$, in order to make them *unrecognisable*.

Second idea

Knot-based Key Exchange II

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.

Knot-based Key Exchange II

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A \# K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.

Second idea

Knot-based Key Exchange II

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.

Second idea

Knot-based Key Exchange II

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.
4. Alice computes $A\#(B\#K) = A\#B\#K$.

Second idea

Knot-based Key Exchange II

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.
4. Alice computes $A\#(B\#K) = A\#B\#K$.
5. Bob computes $B\#(A\#K) = B\#A\#K$.

Knot-based Key Exchange II

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.
4. Alice computes $A\#(B\#K) = A\#B\#K$.
5. Bob computes $B\#(A\#K) = B\#A\#K$.

The secret common key is $A\#B\#K = B\#A\#K$.

Knot-based Key Exchange II

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.
4. Alice computes $A\#(B\#K) = A\#B\#K$.
5. Bob computes $B\#(A\#K) = B\#A\#K$.

The secret common key is $A\#B\#K = B\#A\#K$.

Problem II: $A\#B\#K$ and $B\#A\#K$ are given in different representations.

Knot-based Key Exchange II

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings.
2. Alice chooses a knot A of at most n crossings, computes $A \# K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B \# K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.
4. Alice computes $A \#(B \# K) = A \# B \# K$.
5. Bob computes $B \#(A \# K) = B \# A \# K$.

The secret common key is $A \# B \# K = B \# A \# K$.

Problem II: $A \# B \# K$ and $B \# A \# K$ are given in different representations. We can apply an *invariant* to obtain the same value.

Final idea

Knot-based Key Exchange (final version)

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings and a finite type invariant V.

Final idea

Knot-based Key Exchange (final version)

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings and a finite type invariant V.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.

Final idea

Knot-based Key Exchange (final version)

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings and a finite type invariant V.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.

Final idea

Knot-based Key Exchange (final version)

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings and a finite type invariant V.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.
4. Alice computes $\underline{V}(A\#(B\#K)) = \underline{V}(A\#B\#K)$.

Final idea

Knot-based Key Exchange (final version)

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings and a finite type invariant V.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.
4. Alice computes $\underline{V}(A\#(B\#K)) = \underline{V}(A\#B\#K)$.
5. Bob computes $\underline{V}(B\#(A\#K)) = \underline{V}(B\#A\#K)$.

Final idea

Knot-based Key Exchange (final version)

1. Alice and Bob publicly agree on a positive integer n and a knot K with at most n crossings and a finite type invariant V.
2. Alice chooses a knot A of at most n crossings, computes $A\#K$, applies random Reidemeister moves and sends it to Bob. Her secret key is A.
3. Bob chooses a knot B of at most n crossings, computes $B\#K$, applies random Reidemeister moves and sends it to Alice. His secret key is B.
4. Alice computes $\underline{V}(A\#(B\#K)) = \underline{V}(A\#B\#K)$.
5. Bob computes $\underline{V}(B\#(A\#K)) = \underline{V}(B\#A\#K)$.

The secret common key is $V(A\#B\#K) = V(B\#A\#K)$.

Final idea

Remarks:

[1] https://github.com/denizkutlay/Randomeisterrandomeister, D. Kutluay

Final idea

Remarks:

- Underlying mathematical problem: Given $V(K)$, $V(A\#K)$ and $V(B\#K)$, find $V(A\#B\#K)$.

[1] https://github.com/denizkutluay/Randomeisterrandomeister, D. Kutluay

Final idea

Remarks:

- Underlying mathematical problem: Given $V(K)$, $V(A\#K)$ and $V(B\#K)$, find $V(A\#B\#K)$.
  Related mathematical problem: Given K and $A\#K$, find A (which is unique).

[1] https://github.com/denizkutluay/Randomeisterrandomeister, D. Kutluay

Final idea

Remarks:

- Underlying mathematical problem: Given $V(K)$, $V(A\#K)$ and $V(B\#K)$, find $V(A\#B\#K)$.
  Related mathematical problem: Given K and $A\#K$, find A (which is unique).
- Recall that $(\textbf{oKnots}, \#, \mathcal{U})$ is an *abelian semigroup*. Moreover, $\mathcal{U}$ is the only invertible element.

[1] https://github.com/denizkutluay/Randomeisterrandomeister, D. Kutluay

Final idea

Remarks:

- Underlying mathematical problem: Given $V(K)$, $V(A\#K)$ and $V(B\#K)$, find $V(A\#B\#K)$.
  Related mathematical problem: Given K and $A\#K$, find A (which is unique).
- Recall that $(\mathbf{oKnots}, \#, \mathcal{U})$ is an *abelian semigroup*. Moreover, $\mathcal{U}$ is the only invertible element.
- To apply random Reidemeister moves, we use the program *Randomeister*[1].

[1] https://github.com/denizkutluay/Randomeisterrandomeister, D. Kutluay

Table of Contents

Invariant choice

- Underliyng mathematical problem: Given $V(K)$, $V(A\#K)$ and $V(B\#K)$, find $V(A\#B\#K)$.

Invariant choice

- Underliyng mathematical problem: Given $V(K)$, $V(A\#K)$ and $V(B\#K)$, find $V(A\#B\#K)$.

Some invariants admit a connected-sum formula, i.e.

$$\Phi(K\#K') = \Phi(K)\cdot\Phi(K'),$$

which could solve the problem.

Invariant choice

- Underliyng mathematical problem: Given $V(K)$, $V(A\#K)$ and $V(B\#K)$, find $V(A\#B\#K)$.

Some invariants admit a connected-sum formula, i.e.

$$\Phi(K\#K') = \Phi(K)\cdot\Phi(K'),$$

which could solve the problem.

N.B. Finite type invariants do not have such a formula.

The best attack is a *sort of* brute force attack.

Best attack

The best attack is a *sort of* brute force attack.

1. Compute $A'\#K$ for all knots A' with at most n crossings.

Best attack

The best attack is a *sort of* brute force attack.

1. Compute $A'\#K$ for all knots A' with at most n crossings.
   N.B. It is not enough to just compare $A\#K$ with $A'\#K$ for all K', because the Recognition Problem is hard.

Best attack

The best attack is a *sort of* brute force attack.

1. Compute $A'\#K$ for all knots A' with at most n crossings.
   N.B. It is not enough to just compare $A\#K$ with $A'\#K$ for all K', because the Recognition Problem is hard.
2. Compute $\Phi(A'\#K)$ and compare it to $\Phi(A\#K)$ for all A', where Φ is a fixed *good* invariant.

Best attack

The best attack is a *sort of* brute force attack.

1. Compute $A'\#K$ for all knots A' with at most n crossings.
   N.B. It is not enough to just compare $A\#K$ with $A'\#K$ for all K', because the Recognition Problem is hard.
2. Compute $\Phi(A'\#K)$ and compare it to $\Phi(A\#K)$ for all A', where Φ is a fixed *good* invariant.
   N.B. We do not have complete invariants.

Best attack

The best attack is a *sort of* brute force attack.

1. Compute $A'\#K$ for all knots A' with at most n crossings.
   N.B. It is not enough to just compare $A\#K$ with $A'\#K$ for all K', because the Recognition Problem is hard.
2. Compute $\Phi(A'\#K)$ and compare it to $\Phi(A\#K)$ for all A', where Φ is a fixed *good* invariant.
   N.B. We do not have complete invariants.
3. If you obtain just one correspondence, it is A.

Best attack

The best attack is a *sort of* brute force attack.

1. Compute $A'\#K$ for all knots A' with at most n crossings.
   N.B. It is not enough to just compare $A\#K$ with $A'\#K$ for all K', because the Recognition Problem is hard.
2. Compute $\Phi(A'\#K)$ and compare it to $\Phi(A\#K)$ for all A', where Φ is a fixed *good* invariant.
   N.B. We do not have complete invariants.
3. If you obtain just one correspondence, it is A.
   In general, you will obtain more than one correspondence, so you have to choose *another* invariant and restart.

Choice of parameters

Goal: choose n to reach a 128-bit security level $\rightsquigarrow$ $> 2^{128}$ operations

Polynomial time knot polynomial Z_1 [1, 5] $\rightsquigarrow$ n^6 operations

Choice of parameters

Goal: choose n to reach a 128-bit security level $\rightsquigarrow$ $> 2^{128}$ operations

Polynomial time knot polynomial Z_1 [1, 5] $\rightsquigarrow$ n^6 operations

$$Z_1(K_1 \# K_2) = \Delta_{K_2}^2 Z_1(K_1) + \Delta_{K_1}^2 Z_1(K_2)$$

Choice of parameters

Goal: choose n to reach a 128-bit security level $\rightsquigarrow$ $> 2^{128}$ operations

Polynomial time knot polynomial Z_1 [1, 5] $\rightsquigarrow$ n^6 operations

$$Z_1(K_1 \# K_2) = \Delta^2_{K_2} Z_1(K_1) + \Delta^2_{K_1} Z_1(K_2)$$

Choice of parameters

Goal: choose n to reach a 128-bit security level $\rightsquigarrow$ $> 2^{128}$ operations

Polynomial time knot polynomial Z_1 [1, 5] $\rightsquigarrow$ n^6 operations

Alexander Polynomial Δ_K $\rightsquigarrow$ n^3 operations

$$Z_1(K_1 \# K_2) = {\Delta_{K_2}}^2 Z_1(K_1) + {\Delta_{K_1}}^2 Z_1(K_2)$$

Choice of parameters

Goal: choose n to reach a 128-bit security level $\rightsquigarrow$ $> 2^{128}$ operations

Polynomial time knot polynomial Z_1 [1, 5] $\rightsquigarrow$ n^6 operations

Alexander Polynomial Δ_K $\rightsquigarrow$ n^3 operations

$$Z_1(K_1 \# K_2) = {\Delta_{K_2}}^2 Z_1(K_1) + {\Delta_{K_1}}^2 Z_1(K_2)$$

It is enough to consider $K_1 \# K_2 \# K_3 \# K_4 \# K_5$ with K_i **prime** knots with 19 crossings, since

$$\#\{\text{prime knots with 19 crossings}\} \approx 3 \cdot 10^8$$

$$\Rightarrow \quad n = 95$$

Table of Contents

Future work

Open questions:

Future work

Open questions:

- Find a better invariant.

Future work

Open questions:

- Find a better invariant.
- How many times do we have to apply Reidemester moves to get an equivalent knot that looks as random as possible?

Future work

University of Zurich[UZH]

Open questions:

- Find a better invariant.
- How many times do we have to apply Reidemester moves to get an equivalent knot that looks as random as possible?
- Given a string of quaterns of integers, when it represents an encoded knot?

Future work

Open questions:

- Find a better invariant.
- How many times do we have to apply Reidemester moves to get an equivalent knot that looks as random as possible?
- Given a string of quaterns of integers, when it represents an encoded knot?
- No attempt has yet been made to implement our protocol.

Thanks for your attention!

(Submitted to Cryptology ePrint Archive)

[1] Dror Bar-Natan and Roland van der Veen. "A polynomial time knot polynomial". In: *Proceedings of the American Mathematical Society* 147.1 (2019), pp. 377–397.

[2] Whitfield Diffie and Martin Hellman. "New Directions in cryptography (1976)". In: *IEEE Trans. Inform. Theory* 22 (1976), pp. 644–654.

[3] Mikhail Goussarov, Michael Polyak, and Oleg Viro. "Finite-type invariants of classical and virtual knots". In: *Topology* 39.5 (2000), pp. 1045–1068.

[4] Gérard Maze, Chris Monico, and Joachim Rosenthal. "Public Key Cryptography based on Semigroup Actions". In: *Adv. in Math. of Communications 1.4* (2007), pp. 489–507.

[5] Robert John Quarles. *A New Perspective on a Polynomial Time Knot Polynomial*. Louisiana State University and Agricultural & Mechanical College, 2022.