J. Math. Cryptol. **4** (2010), 149 – 174
DOI 10.1515/JMC.2010.006 © de Gruyter 2010

Mean-set attack: cryptanalysis of Sibert et al. authentication protocol

Natalia Mosina and Alexander Ushakov

Communicated by María Isabel González Vasco

Abstract. We analyze the Sibert et al. group-based (Feige–Fiat–Shamir type) authentication protocol and show that the protocol is not computationally zero-knowledge. In addition, we provide experimental evidence that our approach is practical and can succeed even for groups with no efficiently computable length function such as braid groups. The novelty of this work is that we are not attacking the protocol by trying to solve an underlying complex algebraic problem, namely, the conjugacy search problem, but use a probabilistic approach, instead.

Keywords. Group-based cryptography, zero knowledge, authentication protocol, probability on graphs and groups, braid group, mean-set, mean-set attack principle, shift search problem.

2010 Mathematics Subject Classification. 94A60, 60B15, 68W30, 20F36.

1 Introduction

The group-based cryptography attracted a lot of attention after invention of the Anshel–Anshel–Goldfeld [1] and Ko–Lee et al. [20] key-exchange protocols in 1999. Since then a number of new cryptographic protocols, including public-key *authentication protocols*, based on infinite groups were invented and analyzed. One may consult [24] and [11] to learn more about general group-based cryptography. In this paper we consider a particular interactive group-based authentication scheme, Sibert et al. protocol (see [11, 34]).

Recall that any *interactive proof of knowledge system* is a multi-round randomized protocol for two parties, in which one of the parties (the Prover) wishes to convince another party (the Verifier) of the validity of a given assertion. Every interactive proof of knowledge should satisfy *completeness* and *soundness* properties [14, 16]:

The work of the first author was partially supported by the PSC-CUNY Grant Award 60014-40 41. The work of the second author was partially supported by the NSF grant DMS-0914773.

Completeness: If the assertion is true, it should be accepted by the Verifier with high probability.

Soundness: If the assertion is false, then the Verifier rejects it with high probability.

If the Prover does not trust the Verifier and does not want to compromise any private information in the process of providing the proof of identity, then the following property, concerned with the preservation of security, becomes very important:

Zero-Knowledge (ZK): Except the validity of the Prover's assertions, no other information is revealed in the process of the proof.

If a given protocol possesses the zero-knowledge property, then it is considered to be a *zero-knowledge interactive proof system* [16].

There are three different notions of zero-knowledge that have been commonly used in the literature [11, 15, 16]; namely, perfect zero-knowledge, statistical zero-knowledge, and computational zero-knowledge. The first notion is the most strict definition of ZK, which is rarely useful in practice. The last notion of the ZK property (computational zero-knowledge) is the most liberal notion, and it is used more frequently in practice than the others.

Sibert et al. authentication protocol, is an example of an interactive (dynamic, randomized) proof system. In this paper, we use probabilistic tools, introduced in [27] and outlined in Section 2.3 below, to design an attack on this particular cryptographic primitive and show that it is not computationally zero-knowledge. In addition, we conduct some experiments that support our conclusions and show that the protocol is not secure in practice.

1.1 Description of the protocol

The Sibert's protocol is an iterated two-party three-pass Feige–Fiat–Shamir [14] type authentication protocol. There are two slightly different descriptions of the protocol available in [11] and [34] with two different key generation algorithms. In [34], the protocol is introduced as *Scheme II*. Here, we follow the description of the scheme from the survey [11], except for the minor notational modifications in the conjugation. These modifications do not affect the protocol and its cryptographic properties at all (inverting r and y in [11] would resolve it). In addition, [11] and [34] treat the protocol slightly differently themselves, with and without a collision-free one-way hash function, respectively. Nevertheless, it is not essential for our analysis.

Let G be a (non-commutative, infinite) group, called the *platform group* and μ a probability measure on G. The Prover's *private key* is an element $s \in G$, the Prover's *public key* is a pair (w, t), where w is an arbitrary element of the group G, called *the base element*, and $t = s^{-1}ws$ is a conjugate of w by s. In addition, we assume that H is a collision-free one-way hash function from G to $\{0, 1\}^N$. A single round of the protocol is performed as follows:

(i) The Prover chooses a random element $r \in G$, called the *nonce*, according to the probability measure μ, and sends $x = H(r^{-1}tr)$, called the *commitment*, to the Verifier.

(ii) The Verifier chooses a random bit c, called the *challenge*, and sends it to the Prover.

- If $c = 0$, then the Prover sends $y = r$ to the Verifier and the Verifier checks if the equality $x = H(y^{-1}ty)$ is satisfied.
- If $c = 1$, then the Prover sends $y = sr$ to the Verifier and the Verifier checks if the equality $x = H(y^{-1}wy)$ is satisfied.

This round is repeated k times to guarantee the *soundness error* (i.e., probability that a cheating Prover will be able to convince the Verifier of a false statement) of order 2^{-k}, which is considered to be negligible if k is large, say $k \geq 100$. The Sibert's protocol satisfies both, completeness and soundness, properties of interactive proof systems.

In addition, [34] describes another authentication protocol, the so-called *Scheme III*, which is different from the one described above. Even though techniques of this paper do not directly apply to that protocol, we believe that using similar ideas, this scheme can be successfully attacked as well.

1.2 Security of the protocol

Note that if an intruder (named Eve) can compute the secret element s or any element $s' \in G$ such that $t = s'^{-1}ws'$, i.e., if Eve can solve *the conjugacy search problem* for G, then she can authenticate as the Prover. Thus, as indicated in [34], the computational difficulty of the conjugacy search problem for G is necessary for security of this protocol.

Originally, it was proposed to use braid groups B_n (see [2, 13, 19]) as platform groups, because there was no efficient solution of the conjugacy search problem for B_n known. This motivated a lot of research about braid groups. As a result of recent developments [3–5]), there is an opinion that the conjugacy search problem for B_n can be solved in polynomial time. If that is true in fact, then the Sibert et al. authentication protocol is insecure for B_n. Nevertheless, the same protocol can be used with other platform groups and, hence, it is important to have tools for analysis of this type of general Sibert protocols. We show in the present paper that it is not necessary to solve the conjugacy search problem for G to break the scheme. Instead, one can analyze zero-knowledge property of the protocol by employing ideas from probability theory and show that the protocol is insecure under a mild assumption of existence of an efficiently computable length function for the platform group G. Even for groups with no efficiently computable length function, such as B_n, a reasonable approximation can do the job.

Now, let μ be a probability measure on a platform group G. We say that μ is *left-invariant* if for every $A \subseteq G$ and $g \in G$ the equality $\mu(A) = \mu(gA)$ holds. The following result is proved in [34].

Proposition ([34]). *Let G be a group. If the conjugacy search problem for G is computationally hard (cannot be solved by a probabilistic polynomial time Turing machine) and μ is a left-invariant probability measure on G then the outlined above protocol is a zero knowledge interactive proof system.*

Clearly, there are no left-invariant probability measures on braid groups, used as platform groups in the protocol, and, therefore, as noticed in [11] and [34], this protocol cannot be a perfect zero knowledge interactive proof system when used with an infinite group such as B_n. Nevertheless, it is conjectured in [34] that the scheme can be computationally zero knowledge for certain distributions μ on B_n. The authors supported that conjecture by statistical arguments based on length analysis.

1.3 The idea of mean-set attack: the shift search problem

If we look at the protocol outlined in Section 1.1, we observe that the Prover sends to the Verifier a sequence of random elements of two types: r and sr, where r is a randomly generated element and s is the Prover's secret element. Any passive eavesdropper (Eve) can arrange a table of challenge/response transactions, where each row corresponds to a single round of the protocol, as shown in the table, and obtain two sets of elements, corresponding to $c = 0$ and $c = 1$ respectively: $R_0 = \{r_{i_1}, \ldots, r_{i_k}\}$ and $R_1 = \{sr_{j_1}, \ldots, sr_{j_{n-k}}\}$, where all elements r_i are distributed according to μ, i.e., all these elements are generated by the same random generator. Eve's goal is to recover the secret element s based on the intercepted sequences R_0 and R_1. We call this problem a *shift search problem*.

To explain the idea of the *mean-set attack*, assume for a moment that the group G is an infinite cyclic group $\mathbb{Z}$. In that case, we can rewrite the elements of R_1 in additive notation $\{s + r_{j_1}, \ldots, s + r_{j_{n-k}}\}$. Then we can compute the empirical average $\overline{r}_0 = \frac{1}{k}\sum_{m=1}^{k} r_{i_m}$ of the elements in $R_0 \subset \mathbb{Z}$ and the empirical average $\overline{r}_1 = \frac{1}{n-k}\sum_{l=1}^{n-k}(s + r_{j_l}) = s + \frac{1}{n-k}\sum_{l=1}^{n-k} r_{j_l}$ of the elements in $R_1 \subset \mathbb{Z}$.

Round	Challenge	Response type #1	Response type #2
1	$c = 1$	–	sr_1
2	$c = 0$	r_2	–
3	$c = 0$	r_3	–
4	$c = 1$	–	sr_4
5	$c = 0$	r_5	–
...	...	...	...
n	$c = 0$	r_n	–

By the strong law of large numbers for real-valued random variables the larger the sequence R_0 is, the closer the value of $\overline{r}_0$ to the actual mean $\mathbb{E}(\mu)$ of the distribution μ on $\mathbb{Z}$, induced by r. Similarly, the larger the sequence R_1 is, the closer the value of $\overline{r}_1$ is to the number $s + \mathbb{E}(\mu)$. Therefore, subtracting $\overline{r}_0$ from $\overline{r}_1$, we obtain a good guess of what s is. Observe three crucial properties that allow us to compute the secret element in the case $G = \mathbb{Z}$:

(AV1) (Strong law of large numbers for real-valued random variables) If $\{\xi_i\}_{i=1}^{\infty}$ is a sequence of independent and identically distribute (i.i.d.) real-valued random variables, then

$$\frac{1}{n}\sum_{i=1}^{n}\xi_i \to \mathbb{E}(\xi_1)$$

with probability one as $n \to \infty$, provided $\mathbb{E}(\xi_1) < \infty$.

(AV2) ("Shift" property or linearity) For any real-valued random variable ξ, the formula

$$\mathbb{E}(c + \xi) = c + \mathbb{E}(\xi)$$

holds.

(AV3) (Efficient computations) The average value $\frac{1}{n}\sum_{i=1}^{n}\xi_i$ is efficiently computable.

Geometrically, we can interpret this approach as follows. Given a large sample of random, independent, and identically distributed points $r_{i_1}, \dots, r_{i_k}$ and a large sample of shifted points $s + r_{j_1}, \dots, s + r_{j_{n-k}}$ on the real line, the shift s is "effectively visible".

It turns out that the same is true in general infinite groups. One can generalize a number of mathematical tools of the classical probability theory to finitely generated groups (see [27] and Section 2.3 below) in order to have the counterparts of (AV1), (AV2), and (AV3). Indeed,

- for a random group element $\xi : \Omega \to G$, one can define a set $\mathbb{E}(\xi) \subseteq G$ called the *mean-set*,
- for a sample of n random group elements $\xi_1, \ldots, \xi_n$, one can define their average – a set $\mathbb{S}_n = \mathbb{S}(\xi_1, \ldots, \xi_n) \subseteq G$ called the *sample mean-set* of elements $\xi_1, \ldots, \xi_n$,

so that we have a "shift" property $\mathbb{E}(s\xi) = s\mathbb{E}(\xi)$ and a generalization of the strong law of large numbers (SLLN) for groups with respect to $\mathbb{E}(\xi)$ in a sense that $\mathbb{S}(\xi_1, \ldots, \xi_n)$ converges to $\mathbb{E}(\xi_1)$ as $n \to \infty$ with probability one (see Section 2.3 for precise definitions and statements). In addition, assume that sample mean $\mathbb{S}(\xi_1, \ldots, \xi_n)$ is efficiently computable. Using the operator $\mathbb{S}$, Eve can compute a set

$$\mathbb{S}(sr_{j_1}, \ldots, sr_{j_{n-k}}) \cdot [\mathbb{S}(r_{i_1}, \ldots, r_{i_k})]^{-1},$$

which should contain s with high probability when n is sufficiently large. This is the idea of the mean-set attack and our approach to the *shift search problem.* Furthermore, one can show that the more rounds of the protocol are performed, the more information about the secret key our attack gains (note that at the same time the protocol is iterated by its nature, and large number of rounds is important for its reliability in a sense of the soundness property). The discussion above leads to the main theoretical results of this paper, proved in Section 4.

Theorem A (Mean-set attack principle – I). *Let G be a group, X a finite generating set for G, $s \in G$ a secret fixed element, and $\xi_1, \xi_2, \ldots$ a sequence of randomly generated i.i.d. group elements, such that $\mathbb{E}(\xi_1) = \{g\}$. If $\xi_1, \ldots, \xi_n$ is a sample of random elements of G generated by the Prover, $c_1, \ldots, c_n$ a succession of random bits (challenges) generated by the Verifier, and*

$$y_i = \begin{cases} r_i & \text{if } c_i = 0; \\ sr_i & \text{if } c_i = 1 \end{cases}$$

random elements representing responses of the Prover, then there exists a constant $D = D(G, \mu)$ such that

$$\mathbf{P}(s \notin \mathbb{S}(\{y_i \mid c_i = 1, i = 1, \ldots, n\}) \cdot \mathbb{S}(\{y_i \mid c_i = 0, i = 1, \ldots, n\})^{-1}) \leq \frac{D}{n}.$$

Theorem B (Mean-set attack principle – II). *If, in addition to the assumptions of Theorem A, the distribution μ has finite support, then there exists a constant $D = D(G, \mu)$ such that*

$$\mathbf{P}(s \notin \mathbb{S}(\{y_i \mid c_i = 1, i = 1, \ldots, n\}) \cdot \mathbb{S}(\{y_i \mid c_i = 0, i = 1, \ldots, n\})^{-1}) \leq O(e^{-Dn}).$$

1.4 Outline

Section 2 reviews some necessary graph- and group-theoretic preliminaries that constitute the setting of our work. In Section 2.3, we recall the notion of the mean-set (expectation) of a (graph-)group-valued random element, introduced in [27], and main theorems relevant to this object to prepare the ground for the main results; in particular, we discuss the "shift" property, the strong law of large numbers, and the analogues of Chebyshev and Chernoff-like inequalities for graphs and groups. In Section 3, we propose an algorithm for computing mean-sets. Next, we turn to formulations and proofs of the main theoretical results of this paper, the mean-set attack principles under different assumptions. This task is carried out in Section 4. At the end of that section, we indicate that even if the proposed algorithm fails, we can still gain some information about the secret key of the Prover. In other words, the more rounds of the protocol are performed, the more information about the secret key we can gain. In Section 5.1, we present results of our experiments with the classical key generation according to [11]. Section 5.2 is concerned with results of experiments with the alternative (special) key generation proposed by Sibert et al. in [34]. At the end, in Section 6, we discuss possible methods for defending against the mean-set attack.

2 Preliminaries

Let us briefly recall some definitions of group and graph theory. For a better insight into graph theory, the reader is referred to [35], while [21] can serve as a good introduction into group theory.

2.1 Graphs

An *undirected graph* Γ is an ordered pair of sets (V, E) where

- $V = V(\Gamma)$ is called the *vertex set*;
- $E = E(\Gamma)$ is a set of unordered pairs $(v_1, v_2) \in V \times V$ called the *edge set*.

If $e = (v_1, v_2) \in E$ then we say that v_1 and v_2 are *adjacent* in Γ. The number of vertices adjacent to v is called the *degree* of v. We say that the graph Γ is *locally-finite* if every vertex has a finite degree.

A *directed graph* Γ is an ordered pair of sets (V, E) where $E = E(\Gamma)$ is a set of ordered pairs $(v_1, v_2) \in V \times V$. If $e = (v_1, v_2) \in E$, then we say that v_1 is the *origin* of the edge e, denoted by $o(e)$, and v_2 is the *terminus* of e, denoted by $t(e)$. An undirected graph can be viewed as a directed graph in which a pair $(v_1, v_2) \in E$ serves as two edges (v_1, v_2) and (v_2, v_1).

A *path* p in a directed graph Γ is a finite sequence of edges $e_1, \ldots, e_n$ such that $t(e_i) = o(e_{i+1})$. The vertex $o(e_1)$ is called the *origin* of the path p and is denoted by $o(p)$. The vertex $t(e_n)$ is called the *terminus* of the path p and is denoted by $t(p)$. The number n is called the *length* of the path p and is denoted by $|p|$. We say that two vertices $v_1, v_2 \in V(\Gamma)$ are *connected*, if there exists a path from v_1 to v_2 in Γ. The graph Γ is *connected* if every pair of vertices is connected.

The *distance* between v_1 and v_2 in a graph Γ is the length $d(v_1, v_2)$ of a shortest path between v_1 and v_2. If v_1 and v_2 are disconnected, then $d(v_1, v_2) = \infty$. We say that a path $p = e_1, \ldots, e_n$ from v_1 to v_2 is *geodesic* in a graph Γ if $d(o(p), t(p)) = d(v_1, v_2) = n$, i.e., if p is a shortest path from v_1 to v_2.

A path $p = e_1, \ldots, e_n$ in a graph Γ is *closed*, if $o(p) = t(p)$. In this case we say that p is a *cycle* in Γ. A path p is *simple*, if no proper segment of p is a cycle. The graph Γ is a *tree* if it does not contain a simple cycle.

2.2 Groups and Cayley graphs

Consider a finite set, also called *alphabet*, $X = \{x_1, \ldots, x_n\}$, and let X^{-1} be the set of formal inverses $\{x_1^{-1}, \ldots, x_n^{-1}\}$ of elements in X. This defines an involution $^{-1}$ on the set $X^{\pm 1} := X \cup X^{-1}$ which maps every symbol $x \in X$ to its formal inverse $x^{-1} \in X^{-1}$ and every symbol $x^{-1} \in X^{-1}$ to the original $x \in X$. An alphabet X is called a *group alphabet* if $X^{-1} \subseteq X$, and there is an involution which maps elements of X to their inverses. An X*-digraph* is a graph (V, E) with edges labeled by elements in $X^{\pm 1} = X \cup X^{-1}$ such that for any edge $e = u \overset{x}{\to} v$ there exists an edge $v \overset{x^{-1}}{\to} u$, which is called the inverse of e and is denoted by e^{-1}. See [18] for more information on X-digraphs.

Let G be a group and $X \subset G$ a set of generators for G, i.e. $G = \langle X \rangle$. Assume that X is closed under inversion, i.e., $X = X^{\pm 1}$. The *Cayley graph* $C_G(X)$ of G relative to X is a labeled graph (V, E), where the vertex set is $V = G$, and the edge set E contains all edges of the form $g_1 \overset{x}{\to} g_2$ where $g_1, g_2 \in G$, $x \in X$ and $g_2 = g_1 x$ and only them. The *distance* between elements $g_1, g_2 \in G$ relative to the generating set X is the distance in the graph $C_G(X)$ between vertices g_1 and g_2 or, equivalently,

$$d_X(g_1, g_2) = \min\{n \mid g_1 x_1^{\varepsilon_1} x_2^{\varepsilon_2} \ldots x_n^{\varepsilon_n} = g_2 \text{ for some } x_i \in X, \varepsilon_i = \pm 1\}.$$

2.3 Random (graph-)group elements

In this section, we recall some of the main notions and results of [27] that are employed further in the present paper. Let $\Gamma = (V, E)$ be a locally-finite connected graph and $(\Omega, \mathcal{F}, P)$ a probability space. A measurable mapping $\xi : \Omega \to V(\Gamma)$

is called a *random graph element* defined on a given probability space. A random Γ-element ξ induces an atomic probability measure μ on $V(\Gamma)$ defined in a usual way as

$$\mu(v) = \mu_\xi(v) = \mathbf{P}\{\omega \mid \xi(\omega) = v\}, \quad v \in V(\Gamma).$$

Define a *weight function* $M_\xi : V(\Gamma) \to \mathbb{R}$ by

$$M(v) = M_\xi(v) = \sum_{s \in V(\Gamma)} d^2(v,s)\mu_\xi(s),$$

where $d(v,s)$ is the distance between v and s in Γ. The domain of M is the set

$$\operatorname{domain}(M) = \Big\{v \in V(\Gamma) \mid \sum_{s \in V(\Gamma)} d^2(v,s)\mu_\xi(s) < \infty\Big\}.$$

It is proved in [27] that for any distribution μ on $V(\Gamma)$ either $\operatorname{domain}(M) = \emptyset$ or $\operatorname{domain}(M) = V(\Gamma)$. In the case when $\operatorname{domain}(M) = V(\Gamma)$, we say that $M(\cdot)$ is *totally defined*. Given that $\operatorname{domain}(M) = V(\Gamma)$, the *mean-set* of a Γ-valued ξ is defined to be a set of vertices minimizing the weight function, i.e.,

$$\mathbb{E}(\xi) = \{v \in V(\Gamma) \mid M(v) \le M(u),\ \forall u \in V(\Gamma)\}. \tag{2.1}$$

Sometimes we write $\mathbb{E}(\mu)$ and speak of the mean-set of distribution μ. Using the Cayley graph construction one can similarly define a notion of the mean-set for a finitely generated group G (relative to a fixed generating set). Similar mean values (in different settings) are used rather often; see [27] for some history and literature sources. Below, we recall some results proved in [27].

Lemma 2.1 ([27]). *Let ξ be a random Γ-element, where Γ is a connected locally-finite graph, with totally defined weight function $M_\xi(\cdot)$. Then the mean-set $\mathbb{E}(\xi)$ is non-empty and finite.*

The next property is an analogue of the property $\mathbb{E}(c + \xi) = c + \mathbb{E}(\xi)$ for real-valued random variables.

Proposition 2.2 (Shift property, [27]). *Let $G = \langle X \rangle$ be a finitely generated group and $g \in G$. Let ξ be a random G-element. Then for a random element ξ_g defined by $\xi_g(\omega) := g\xi(\omega)$ we have $\mathbb{E}(\xi_g) = g\mathbb{E}(\xi)$.*

It is easy to see that this property follows from the fact that for any $g_1, g_2, s \in G$ the equality $d_X(g_1, g_2) = d_X(sg_1, sg_2)$ holds, where $d_X(g_1, g_2)$ is the *distance* between elements $g_1, g_2 \in G$ relative to X (see Section 2).

Now let $\xi_1, \ldots, \xi_n$ be a sample of independent and identically distributed graph-valued random elements $\xi_i : \Omega \to V(\Gamma)$ defined on a given probability space $(\Omega, \mathcal{F}, \mathbf{P})$ and $\mu_n(v)$ be the relative frequency

$$\mu_n(v) = \mu_n(v, \omega) = \frac{|\{i \mid \xi_i(\omega) = v, 1 \le i \le n\}|}{n}$$

with which the value $v \in V(\Gamma)$ occurs in the random sample $\xi_1(\omega), \ldots, \xi_n(\omega)$. Let

$$M_n(v) = \sum_{i \in V(\Gamma)} d^2(v, i) \mu_n(i)$$

be the random weight, called the *sampling weight*, corresponding to $v \in V(\Gamma)$, and $M_n(\cdot)$ the resulting random *sampling weight function*. The set of vertices

$$\mathbb{S}_n = \mathbb{S}(\xi_1, \ldots, \xi_n) = \{v \in V(\Gamma) \mid M_n(v) \le M_n(u), \forall u \in V(\Gamma)\}$$

is called the *sample mean-set* (or *sample center-set*) relative to ξ. The next theorem shows that the sets $\mathbb{S}_n$ and $\mathbb{E}(\xi)$ in Γ play roles analogous to the classical average of real values $\frac{x_1+\cdots+x_n}{n}$ and the classical expectation $\mathbb{E}$ of a real-valued random variable respectively, in the non-commutative case. In other words, the strong law of large numbers generalized to graphs and groups states that our (empirical) sample mean-set $\mathbb{S}_n$ converges to the (theoretical) mean-set $\mathbb{E}(\xi)$ as $n \to \infty$.

Theorem 2.3 (Strong law of large numbers, [27]). *Let Γ be a locally-finite connected graph and $\{\xi_i\}_{i=1}^{\infty}$ a sequence of i.i.d. random Γ-elements. If the weight function $M_{\xi_1}(\cdot)$ is totally defined and $\mathbb{E}(\xi_1) = \{v\}$ for some $v \in V(\Gamma)$, then*

$$\lim_{n \to \infty} \mathbb{S}_n = \mathbb{E}(\xi_1)$$

with probability one.

Similar result holds for multi-vertex mean-sets. See [27] for technical conditions needed, as well as other details. The simplest version of multi-vertex SLLN in terms of *limsup* is as follows:

Theorem 2.4 (Multi-Vertex SLLN, [27]). *Let Γ be a locally-finite connected graph and $\{\xi_i\}_{i=1}^{\infty}$ be a sequence of i.i.d. random Γ-elements. Assume that the weight function $M_{\xi_1}(\cdot)$ is totally defined and $\mathbb{E}(\xi) = \{v_1, \ldots, v_k\}$, where $k \ge 4$. If $\mathbb{E}(\xi_1) \subseteq \mathrm{supp}(\mu)$ then*

$$\limsup_{n \to \infty} \mathbb{S}_n = \mathbb{E}(\xi_1)$$

holds with probability one.

Moreover, the following asymptotic upper bounds (analogues of the classical Chebyshev and Chernoff bounds) on convergence rate hold:

Theorem 2.5 (Chebyshev's inequality for graphs, [27]). *Let Γ be a locally-finite connected graph and $\{\xi_i\}_{i=1}^{\infty}$ a sequence of i.i.d. random Γ-elements. If the weight function $M_{\xi_1}(\cdot)$ is totally defined then there exists a constant $C = C(\Gamma, \xi_1) > 0$ such that*

$$\mathbf{P}(\mathbb{S}(\xi_1, \dots, \xi_n) \not\subseteq \mathbb{E}(\xi_1)) \leq \frac{C}{n}. \tag{2.2}$$

With an additional assumption on μ, we can get even Chernoff-like asymptotic bound.

Theorem 2.6 (Chernoff-like bound for graphs, [27]). *Let Γ be a locally-finite connected graph and $\{\xi_i\}_{i=1}^{\infty}$ a sequence of i.i.d. random Γ-elements. If the weight function $M_{\xi_1}(\cdot)$ is totally defined and μ_{ξ_1} has finite support, then for some constant $C > 0$*

$$\mathbf{P}(\mathbb{S}(\xi_1, \dots, \xi_n) \not\subseteq \mathbb{E}(\xi_1)) \leq O(e^{-Cn}). \tag{2.3}$$

3 Effective computation of a mean-set

Let G be a group and $\{\xi_i\}_{i=1}^{n}$ a sequence of random i.i.d. elements taking values in G such that the corresponding weight function $M(\cdot)$ is totally defined. In Section 2.3, we introduced a notion of the mean-set of ξ that satisfies the desirable properties (AV1) and (AV2) of Section 1.3. One of the technical difficulties encountered in practice is that, unlike the classical average value $(x_1 + \cdots + x_n)/n$ for real-valued random variables, the sample mean-set $\mathbb{S}_n$ is hard to compute. In other words, in general, our definition of the meat-set might not satisfy the property (AV3).

Several problems arise when trying to compute $\mathbb{S}_n$:

- Straightforward computation of the set $\{M(g) \mid g \in G\}$ requires at least $O(|G|^2)$ steps. This is computationally infeasible for large groups G, and impossible for infinite groups. Hence we might want to reduce the search of a minimum to some small part of G.

- There exist infinite groups in which the distance function $d(\cdot,\cdot)$ is very difficult to compute. The braid group B_∞ is an example for such a group. The computation of the distance function for B_∞ is known to be NP-hard, see [31]. Such groups require special treatment. Moreover, there exist infinite groups for which the distance function $d(\cdot,\cdot)$ is not computable. We omit consideration of such groups.

We devise a heuristic procedure to solve the first problem. As proved in [27], if the weight function $M(\cdot)$ satisfies certain local monotonicity properties, then our procedure achieves the desired result. Our algorithm is a simple direct descent heuristic, in which we use the sample weight function M_n that comes from a sample of random group elements $\{g_1, \dots, g_n\}$ from a finitely-generated group G.

Algorithm 3.1 (Direct Descent Heuristic).
INPUT: A group G with a finite set of generators $X \subseteq G$ and a sequence of elements $\{g_1, \dots, g_n\}$ in G.
OUTPUT: An element $g \in G$ that locally minimizes $M_n(\cdot)$.
COMPUTATIONS:

```
A. Choose a random $g \in G$ according to some probability measure $\nu$ on $G$.
B. If for every $x \in X^{\pm 1}$, $M_n(g) \le M_n(gx)$, then output $g$.
C. Otherwise put $g \leftarrow gx$, where $x \in X^{\pm 1}$ is an element minimizing the value
   of $M_n(gx)$ and go to step B.
```

As any other direct descend heuristic method, Algorithm 3.1 might not work if the function M_n has local minima. It is proved in [27] that it always works for trees and, hence, for free groups.

Theorem 3.2 ([27]). *Let μ be a distribution on a locally-finite tree T such that a function M is totally defined. Then Algorithm* 3.1 *for T and M finds a central point (mean-set) of μ on T.*

The second problem of computing $\mathbb{S}_n$ concerns practical computations of length function in G. It turns out that we need a relatively mild assumption to deal with it – the existence of an efficiently computable distance function $d_X(\cdot, \cdot)$; even a "reasonable" approximation of the length function may work. In this work we approximate geodesic length using the method described in [25]. Even though it does not guarantee the optimal result, it was proved to be practically useful in a series of attacks, see [22, 23, 29, 30].

4 The mean-set attack

In this section, we use theoretical results stated above to attack the Sibert et al. protocol, described in Section 1.1. In the following heuristic attack we use the Algorithm 3.1 to compute sample mean-set $\mathbb{S}_n$.

Algorithm 4.1 (The mean-set attack).
INPUT: The Prover's public element (t, w) and sequences R_0 and R_1 as in the protocol.
OUTPUT: An element z satisfying the equality $t = z^{-1}wz$ (which can be considered as the Prover's private key), or *Failure*.
COMPUTATIONS:

A. Apply Algorithm 3.1 to R_0 and obtain g_0.

B. Apply Algorithm 3.1 to R_1 and obtain g_1.

C. If $g_1 g_0^{-1}$ satisfies $t = (g_1 g_0^{-1})^{-1} w (g_1 g_0^{-1})$ then output $g_1 g_0^{-1}$. Otherwise output *Failure*.

If the algorithm outputs an element $z \in G$, then z can serve as the Prover's original secret s; any solution of the conjugacy equation $t = x^{-1}wx$ does. In general, z can be different from s, and there are no means for the adversary to determine whether $z = s$. In spite of that, Eve, who is only trying to authenticate as the Prover, considers this z a success. On the other hand, since our goal is to show that the protocol is not computationally zero-knowledge, we estimate the probability to find s. Only this original secret element s is considered as a success in our analysis. Other outcomes that work for Eve (when $z \neq s$) are ignored.

The theorems below give asymptotic bounds on the failure rate (for the original s) in the mean-set attack. We show that the probability of the failure can decrease linearly or exponentially, depending on the distribution μ.

Theorem 4.2 (Mean-set attack principle – I). *Let G be a group, X a finite generating set for G, $s \in G$ a secret fixed element, and $\xi_1, \xi_2, \dots$ a sequence of randomly generated i.i.d. group elements, such that $\mathbb{E}(\xi_1) = \{g\}$. If $\xi_1, \dots, \xi_n$ is a sample of random elements of G generated by the Prover, $c_1, \dots, c_n$ a succession of random bits (challenges) generated by the Verifier, and*

$$y_i = \begin{cases} r_i & \text{if } c_i = 0; \\ s r_i & \text{if } c_i = 1 \end{cases}$$

random elements representing responses of the Prover, then there exists a constant D such that

$$\mathbf{P}(s \notin \mathbb{S}(\{y_i \mid c_i = 1, i = 1, \dots, n\}) \cdot \mathbb{S}(\{y_i \mid c_i = 0, i = 1, \dots, n\})^{-1}) \le \frac{D}{n}.$$

Proof. It follows from Theorem 2.5 that there exists a constant C such that

$$\mathbf{P}(\mathbb{S}(\{y_i \mid c_i = 0, i = 1, \dots, n\}) \neq \{g\}) \le \frac{C}{|\{i \mid c_i = 0, i = 1, \dots, n\}|}.$$

Applying Chebyshev's inequality to Bernoulli random variables $\{c_i\}$ having $\mathbb{E}(c_i) = \frac{1}{2}$ and $\sigma^2_{c_i} = \frac{1}{4}$, we obtain

$$\mathbf{P}\Big(|\{i \mid c_i = 0, i = 1, \ldots, n\}| < \frac{n}{4}\Big) < \frac{4}{n}.$$

In more detail, if the number of zeros in our sample of challenges is less than $\frac{n}{4}$, then the number of ones is greater or equal to $\frac{3n}{4}$, and we have

$$\mathbf{P}\Big(|\{i \mid c_i = 0, i = 1, \ldots, n\}| < \frac{n}{4}\Big) < \mathbf{P}\Big(\Big|\sum_{i=1}^{n} c_i - \frac{n}{2}\Big| \geq \frac{n}{4}\Big).$$

Note that

$$\Big|\sum_{i=1}^{n} c_i - \frac{n}{2}\Big| \geq \frac{n}{4} \Leftrightarrow \Big|\frac{\sum_{i=1}^{n} c_i}{n} - \frac{1}{2}\Big| \geq \frac{1}{4}$$

and

$$\mathbf{P}\Big(\Big|\frac{\sum_{i=1}^{n} c_i}{n} - \frac{1}{2}\Big| \geq \frac{1}{4}\Big) \leq \frac{4}{n}$$

from the classical Chebyshev inequality for sample means with $\varepsilon = \frac{1}{4}$.

It follows that

$$\mathbf{P}(\mathbb{S}(\{y_i \mid c_i = 0, i = 1, \ldots, n\}) \neq \{g\}) \leq \frac{4}{n} + \frac{4C}{n} \leq \frac{4+4C}{n}.$$

Similarly, we prove that $\mathbf{P}(\mathbb{S}(\{y_i \mid c_i = 1, i = 1, \ldots, n\}) \neq \{sg\}) \leq \frac{4+4C}{n}$. Hence,

$$\mathbf{P}(s \notin \mathbb{S}(\{y_i \mid c_i = 1, i = 1, \ldots, n\}) \cdot \mathbb{S}(\{y_i \mid c_i = 0, i = 1, \ldots, n\})^{-1}) \leq \frac{8+8C}{n}. \quad \square$$

Furthermore, we can get Chernoff-like asymptotic bound if we impose one restriction on distribution μ. Recall the original Hoeffding's inequality [17] well known in probability theory. Assume that $\{x_i\}$ is a sequence of independent random variables and that every x_i is almost surely bounded, i.e., $\mathbf{P}(x_i - \mathbb{E}(x_i) \in [a_i, b_i]) = 1$ for some $a_i, b_i \in \mathbb{R}$. Then for the sum $S_n = x_1 + \cdots + x_n$, the inequality

$$\mathbf{P}(S_n - \mathbb{E}(S_n) \geq n\varepsilon) \leq \exp\Big(-\frac{2n^2\varepsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\Big)$$

holds. If x_i are identically distributed, then we get the inequality

$$\mathbf{P}\left(\frac{1}{n}(x_1 + \cdots + x_n) - \mathbb{E}(x_1) \geq \varepsilon\right) \leq 2\exp\left(-\frac{2\varepsilon^2}{(b-a)^2}n\right). \tag{4.1}$$

Now we can prove the Mean-set attack principle with exponential bounds.

Theorem 4.3 (Mean-set attack principle – II). *Let G be a group, X a finite generating set for G, $s \in G$ a secret fixed element, and $\xi_1, \xi_2, \ldots$ a sequence of randomly generated i.i.d. group elements, such that $\mathbb{E}(\xi_1) = \{g\}$. If $\xi_1, \ldots, \xi_n$ is a sample of random elements of G generated by the Prover, $c_1, \ldots, c_n$ a succession of random bits (challenges) generated by the Verifier,*

$$y_i = \begin{cases} r_i & \text{if } c_i = 0; \\ sr_i & \text{if } c_i = 1 \end{cases}$$

random elements representing responses of the Prover, and the distribution μ has finite support, then there exists a constant $D = D(G, \mu)$ such that

$$\mathbf{P}(s \notin \mathbb{S}(\{y_i \mid c_i = 1, i = 1, \ldots, n\}) \cdot \mathbb{S}(\{y_i \mid c_i = 0, i = 1, \ldots, n\})^{-1}) \\ \leq O(e^{-Dn}).$$

Proof. It follows from Theorem 2.6 that there exists a constant C such that

$$\mathbf{P}(\mathbb{S}(\{y_i \mid c_i = 0, i = 1, \ldots, n\}) \neq \{g\}) \leq O(e^{-C|\{i|c_i=0, i=1,\ldots,n\}|}).$$

Applying inequality (4.1) to Bernoulli random variables $\{c_i\}$, we get

$$\mathbf{P}\left(\sum_{i=1}^{n} c_i - \frac{1}{2} > \frac{1}{4}\right) < e^{-n/8}.$$

Thus, we obtain a bound

$$\mathbf{P}(\mathbb{S}(\{y_i \mid c_i = 0, i = 1, \ldots, n\}) \neq \{g\}) \leq e^{-n/8} + O(e^{-Cn/4}).$$

Similarly, we prove that $\mathbf{P}(\mathbb{S}(\{y_i \mid c_i = 1, i = 1, \ldots, n\}) \neq \{sg\}) \leq e^{-n/8} + O(e^{-Cn/4})$. Hence,

$$\mathbf{P}(s \notin \mathbb{S}(\{y_i \mid c_i = 1, i = 1, \ldots, n\}) \cdot \mathbb{S}(\{y_i \mid c_i = 0, i = 1, \ldots, n\})^{-1}) \\ \leq O(e^{-Dn})$$

where $D = \min\{1/8, C/4\}$. □

Algorithm 4.1 can fail. Nevertheless the pair of the obtained elements g_0, g_1 often encodes some additional information about the secret s. Indeed, assume that $\mathbb{E}(\mu) = \{g\}$. The element g_0 obtained at step A of Algorithm 4.1 can be viewed as a product ge_0 for some $e_0 \in G$. Similarly, the element g_1 can be viewed as a product sge_1 for some $e_1 \in G$. Hence Algorithm 4.1 outputs the secret element s whenever $g_1 g_0^{-1} = sge_1 e_0^{-1} g^{-1} = s$, i.e., whenever $e_1 e_0^{-1} = 1$.

Now, assume that Algorithm 4.1 has failed, i.e., $e_1 e_0^{-1} \neq 1$. In this case, one can try to reconstruct the secret element s as a product

$$g_1 \cdot e \cdot g_0^{-1} = sge_1 \cdot e \cdot e_0^{-1} g^{-1}$$

where e is an unknown element of the platform group. Clearly, e gives a correct answer if and only if $e_1 \cdot e \cdot e_0^{-1} = 1$ or $e = e_1^{-1} e_0$. The element

$$e_1^{-1} e_0 \tag{4.2}$$

is called *the error of the method*. Clearly, one only needs to enumerate all words e of length up to $|e_1^{-1} e_0|$ to reconstruct the required s in the form $g_1 e g_0^{-1}$. If a secret element s is chosen uniformly as a word of length l and $|e_1^{-1} e_0| < l$, then we gain some information about s, since the search space for s reduces. We can improve Algorithm 4.1 by adding such enumeration step as follows.

Algorithm 4.4 (The attack – 2).
INPUT: The Prover's public element (t, w). Sequences R_0 and R_1 as in the protocol. The number $k \in \mathbb{N}$ – the expected length of error element $e_1 e_0^{-1}$.
OUTPUT: An element z satisfying the equality $t = z^{-1} w z$ (which can be considered as the Prover's private key), or *Failure*.
COMPUTATIONS:

A. Apply Algorithm 3.1 to R_0 and obtain g_0.

B. Apply Algorithm 3.1 to R_1 and obtain g_1.

C. For every word e of lengths up to k, check if $g_1 e g_0^{-1}$ satisfies the equality $t = (g_1 e g_0^{-1})^{-1} w (g_1 e g_0^{-1})$ and if so output $g_1 e g_0^{-1}$. Otherwise output *Failure*.

5 Experiments

To demonstrate the practical use of our mean-set attack, we perform a series of experiments, which we describe below. In [34], [11] two different methods of generation of nonce elements were proposed, both with the same platform group B_n, which has the following (Artin's) presentation:

$$B_n = \left\langle \sigma_1, \ldots, \sigma_{n-1} \;\middle|\; \begin{array}{ll} \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j & \text{if } |i-j| = 1 \\ \sigma_i\sigma_j = \sigma_j\sigma_i & \text{if } |i-j| > 1 \end{array} \right\rangle.$$

We distinguish between the two ways, classical ([11]) and alternative ([34]), to generate elements of the underlying group by performing two different sets of experiments outlined below in Sections 5.1 and 5.2. In both cases, we observe that the secret information of the Prover is not secure, and the probability to break the protocol grows as the number of rounds of the protocol increases. All experiments are done using the CRAG software package [9].

5.1 Classical key generation

Classical key generation of the elements of B_n was suggested in [11] with parameters $n = 50$ (rank of the braid group) and the lengths of private keys $L = 512$. The length function relative to the Artin generators $\{\sigma_1, \ldots, \sigma_{n-1}\}$ is NP-hard. That is why in this paper, as it was already mentioned in Section 3, we use the approximation of geodesic length method, proposed in [23]. See [22, 23, 29, 30] for a series of successful attacks using this method. We want to emphasize that we compute the sampling weight values in the Algorithm 3.1, which is a subroutine in Algorithm 4.1, using the approximated distance function values in B_n.

One of the disadvantages of the approximation algorithm that we used is that there is no polynomial time upper bound for that as it uses Dehornoy handle-free forms [10]. As a result we do not know the complexity of our algorithm and we do not know how our algorithm scales with parameter values. In each experiment we randomly generate an instance of the authentication protocol and try to break it, i.e., find the private key, using the techniques developed in this paper. Recall that each authentication is a series of k 3-pass commitment-challenge-response rounds. Therefore, an instance of authentication consists of k triples (x_i, c_i, r_i), $i = 1, \ldots, k$ obtained as described in Section 1.1. Here x_i is a commitment, c_i is a challenge, and r_i is a response. A random bit c_i is chosen randomly and uniformly from the set $\{0, 1\}$. In our experiments we make an assumption that exactly half of c_i's are 0 and half are 1. This allows us to see an instance of the protocol as a pair of equinumerous sets $R_0 = \{r_1, \ldots, r_{k/2}\} \subset B_n$ and $R_1 = \{sr'_1, \ldots, sr'_{k/2}\} \subset B_n$.

The main parameters for the system are the rank n of the braid group, the number of rounds k in the protocol, and the length L of secret keys. We generate a single instance of the problem with parameters (n, k, L) as follows:

- A braid s is chosen randomly and uniformly as a word of length L over a group alphabet $\{\sigma_1, \ldots, \sigma_{n-1}\}$. This braid is a secret element which is used only to generate further data and to compare the final element to.
- A sequence $R_0 = \{r_1, \ldots, r_{k/2}\}$ of braid words chosen randomly and uniformly as words of length L over a group alphabet $\{\sigma_1, \ldots, \sigma_{n-1}\}$.
- A sequence $R_1 = \{sr'_1, \ldots, sr'_{k/2}\}$ of braid words, where r'_i are chosen randomly and uniformly as words of length L over a group alphabet $\{\sigma_1, \ldots, \sigma_{n-1}\}$.

For every parameter set (n, k, L) we generate 1000 random instances (R_0, R_1) and run Algorithm 4.1 which attempts to find the secret key s used in the generation of R_1.

Below we present the results of actual experiments done for groups B_5, B_{10}, and B_{20}. Horizontally we have increasing number of rounds k from 10 to 320 and vertically we have increasing lengths L from 10 to 100. Every cell contains a pair $(P\%, E)$ where P is a success rate and E is an average length of the error (4.2) of the method for the corresponding pair (L, k) of parameter values. All experiments were performed using CRAG library [9]. The library provides an environment to test cryptographic protocols constructed from non-commutative groups, for example the braid group.

$L\backslash k$	**10**	**20**	**40**	**80**	**160**	**320**
10	(19%, 1.3)	(72%, 0.3)	(97%, 0.04)	(100%, 0)	(100%, 0)	(100%, 0)
50	(2%, 13.4)	(8%, 9)	(68%, 1.3)	(93%,0.1)	(100%, 0)	(100%, 0)
100	(0%, 53.7)	(0%, 48.1)	(6%, 26.9)	(44%, 14)	(65%, 14.7)	(87%, 5)

Table 1. Experiments in B_5.

$L\backslash k$	**10**	**20**	**40**	**80**	**160**	**320**
10	(15%, 1.8)	(68%, 0.3)	(98%, 0)	(100%, 0)	(100%, 0)	(100%, 0)
50	(0%, 4.5)	(23%, 1.3)	(82%, 0)	(97%, 0)	(99%, 0)	(100%, 0)
100	(1%, 41)	(7% ,23.5)	(33%,5)	(79%, 1)	(97%, 0.6)	(98%, 1.1)

Table 2. Experiments in B_{10}.

$L\backslash k$	**10**	**20**	**40**	**80**	**160**	**320**
10	(15%, 1.6)	(87%, 0.1)	(100%, 0)	(100%, 0)	(100%, 0)	(100%, 0)
50	(0%, 5.4)	(23%, 1.7)	(81%, 0.2)	(100%, 0)	(100%, 0)	(100%, 0)
100	(0%,7.8)	(15%, 2)	(72%, 0.3)	(97%, 0)	(100%, 0)	(100%, 0)

Table 3. Experiments in B_{20}.

We immediately observe from the data in Tables 1–3 that:

- the success rate increases as the number of rounds (sample size) increases;
- the success rate decreases as the length of the key increases;
- the success rate increases as the rank of the group increases;
- the average error length decreases as we increase the number of rounds.

The first observation is the most interesting since the number of rounds is one of the main reliability parameters of the protocol, namely, the soundness error decreases as $1/2^k$ as the number of rounds k gets larger. But, at the same time, we observe that security of the scheme decreases as k increases. The second observation can be interpreted as follows – the longer the braids are the more difficult it is to compute the approximation. The third observation is easy to explain. The bigger the rank of the group the more braid generators commute and the simpler random braids are.

5.2 Alternative key generation

As we have mentioned in Section 1.2, the Sibert et al. scheme, proposed in [34], does not possess perfect zero knowledge property. Nevertheless, the authors of [34] try to achieve computational zero knowledge by proposing a special way of generating public and private information. They provide some statistical evidence that the scheme can be computationally zero knowledge if this alternative key generation is used. In this section we, firstly, outline the proposed key generation method and, secondly, present actual experiments supporting our theoretical results even for this special key generation method.

The method of generating of braids in [34] can be translated to the notation of the present paper as follows. The Prover generates

- nonce elements r as products of L uniformly chosen permutation braids p_i (see [13]) from B_n
$$r = p_1 \cdots p_L,$$
in particular, r belongs to the corresponding positive monoid.
- the secret key s as the inverse of a product of L uniformly chosen *permutation braids* from B_n, i.e.,
$$s = p_1^{-1} \cdots p_L^{-1}.$$

We made a very useful observation when doing the experiments with so generated nonce elements r. We observed that the mean-set in this case is often a singleton set of the form $\{\Delta^k\}$, where Δ is a half-twist braid and $k \in \mathbb{N}$. Therefore, to enhance the performance of Algorithm 3.1 in step B, we test not only generators $x \in X^{\pm 1}$, but also $x = \Delta$, and if (in step C) Δ minimizes the value of $M_n(gx)$, then we put $x \to x\Delta$ and return to step B.

In fact it is an interesting question if the uniform distribution on a sphere in a Garside monoid G^+ has a singleton mean set $\{\Delta_{G^+}^k\}$ for some $k \in \mathbb{N}$, where Δ_{G^+} is the Garside element, Δ, in G^+? This is clearly true for free abelian monoids. As we mention above, experiments show that the same can be true in the braid monoid.

Below we present the results of actual experiments done for the group B_{10}. Horizontally we have increasing number of rounds k from 10 to 320 and vertically we have increasing lengths L (in permutation braids) from 3 to 10. Every cell contains a pair $(P\%, E)$ where P is a success rate and E is the average length of the error for the corresponding pair (L, k) of parameter values.

Since the average Artin length (denoted L' in the tables below) of a permutation braid on n strands is of order n^2, the length of nonce elements grows very fast with L; it is shown in the leftmost column of the tables in parentheses. For instance, we can see that for B_{10} the average length of a product of $L = 3$ permutation braids is 81, the average length of a product of $L = 5$ permutation braids is 138, etc.

L(L')\k	**10**	**20**	**40**	**80**	**160**	**320**
3 (81)	(0%, 24.6)	(0%, 22.5)	(1%, 19.6)	(4%, 16)	(7%, 13.1)	(25%, 12.3)
5 (138)	(0%, 46.7)	(0%, 40.9)	(0%, 32.5)	(2%, 23.3)	(10%, 17.6)	(28%, 14.2)
10 (274)	(0%, 110.2)	(0%, 102.6)	(0%, 103.5)	(0%, 96.3)	(0%, 92.7)	(0%, 87.9)

Table 4. Success rate and average length of the error for experiments in B_{10}.

Again, we observe that success rate increases as we increase the number of rounds, and the average error length decreases as we increase the number of rounds.

6 Defending against the attack

In this section, we describe several principles one can follow in order to defend against the mean-set attack presented in this paper or, at least, to make it computationally infeasible. Defending can be done through a special choice of the platform group G or a special choice of a distribution μ on G. Another purpose of this section is to motivate further study of distributions on groups and computational properties of groups.

6.1 Groups with no efficiently computable length functions

One of the main tools in our technique is an efficiently computable function $d_X(\cdot,\cdot)$ on G. To prevent the attacker from computing mean-sets, one can use a platform group G with a hardly computable length function $d_X(\cdot,\cdot)$ relative to any "reasonable" finite generating set X. By reasonable generating set we mean a set, which is small relative to the main security parameter. Examples of such groups exist. For instance, length function for any finitely presented group with unsolvable word problem is not computable. On the other hand, it is hard to work with such groups, as they do not have efficiently computable normal forms.

A more interesting example is a multiplicative group of a prime field $\mathbb{Z}_p^*$. The group $\mathbb{Z}_p^*$ is cyclic, i.e., $\mathbb{Z}_p^* = \langle a \rangle$ for some primitive root a of p. It is easy to see that the length of an element $b \in \mathbb{Z}_p^*$ satisfies

$$|b| = \begin{cases} \log_a b & \text{if } \log_a b \le (p-1)/2, \\ p - 1 - \log_a b & \text{otherwise,} \end{cases}$$

and hence the problem of computing the length of an element and the discrete logarithm problem are computationally equivalent. The discrete logarithm problem is widely believed to be computationally hard and is used as a basis of security of many cryptographic protocols, most notably the ElGamal [12] and Cramer–Shoup [8] cryptosystems. In other words, $\mathbb{Z}_p^*$ is another example of a group with hardly computable length function.

6.2 Systems of probability measures

Let G be a platform group. Recall that our assumption was that the Prover uses a fixed distribution on the set of nonce elements, i.e., every element r_i is generated using the same random generator. Instead he can use a sequence of probability measures $\{\mu_i\}_{i=1}^{\infty}$, where each measure μ_i, $i = 1, 2, \ldots$, is not used more than once (ever), i.e., every nonce r_i, $i = 1, 2, \ldots$, is generated using a unique distribution $\{\mu_i\}$. In this case, the attacker does not have theoretical grounds for working with sampling mean-sets. Nevertheless, it can turn out that the sequence of random elements $r_1, r_2, \ldots$ can have some other distribution μ^* and the attack will work. Another difficulty with implementing this idea is that there is no systematic study of distributions on general finitely generated groups and, in particular, braid groups. So, it is hard to propose some particular sequence of probability distributions. Some aspects of defining probability measures on infinite groups are discussed in [7] and [6].

6.3 Undefined mean-set

Another way to foil the attack is to use a distribution μ on G such that $\mathbb{E}(\mu)$ is not defined, i.e., the corresponding weight function is not totally defined. In that case the assumption of Theorem 4.2 fails, and it is easy to see that the sampling weights $M_n(g)$ tend to ∞ with probability 1. Nevertheless, we still can compare the sampling weight values, as explained in [26] and [28], where it is shown that the condition of finiteness of $M^{(2)}$ can be relaxed to that of finiteness of $M^{(1)}$. If $M^{(1)}$ is not defined then that means that the lengths of commitments are too large and are impractical.

6.4 Large mean-set

Also, to foil the attack one can use a distribution μ on G such that the set $\mathbb{E}(\mu)$ is large. As an example consider an authentication protocol in [33], based on the difficulty of computing discrete discrete logarithms in groups of prime order. The space of nonce elements in [33] is an additive group $\mathbb{Z}_q$ acting by exponentiations on a bigger group $\mathbb{Z}_p^*$. It is easy to compute length in $(\mathbb{Z}_q, +) = \langle 1 \rangle$. But, since the nonce elements $r \in \mathbb{Z}_q$ are chosen uniformly, it follows that the mean-set is the whole group $\mathbb{Z}_q$ (the uniform measure is right-invariant) and in this case it is impossible to detect the shift s and the mean-set attack fails. We also refer to [32] for a modification of [33] where nonce elements are not taken modulo q and security proof requires a boundary on the number of times the same key is used.

Now, let G be an infinite group. It is impossible to generate elements of G uniformly, but one can try to achieve the property described below that can foil the mean-set attack. Choose a probability measure μ on G so that the mean-set set $\mathbb{E}(\mu)$ is large. Recall that Algorithm 4.1 can find up to one element of G minimizing the weight function. For that it uses Algorithm 3.1 which randomly (according to some measure ν) chooses an element of $g \in G$ and then gradually changes it (descends) to minimize its M value. This way the distribution ν on the initial choices $g \in G$ defines a distribution ν_μ^* on the set of local minima of M on G. More precisely, for $g' \in G$,

$$\nu_\mu^*(g') = \mu\{g \in G \mid \text{Algorithm 3.1 stops with the answer } g' \text{ on input } g\}.$$

Denote by μ_s the *shifted probability measure* on G by an element s defined by $\mu_s(g) = \mu(s^{-1}g)$. If $S \subseteq G$ is the set of local minima of the weight function M relative to μ then the set sS is the set of local minima relative to μ_s. But the distribution $\nu_{\mu_s}^*$ does not have to be induced from ν_μ^* by the shift s, i.e., the equality $\nu_{\mu_s}^*(g) = \nu_\mu^*(s^{-1}g)$ does not have to hold. In fact, the distributions ν_μ^* and $\nu_{\mu_s}^*$

can “favor” unrelated subsets of S and sS respectively. That would definitely foil the attack presented in this paper. On the other hand, if ν_{μ}^{*} and $\nu_{\mu_s}^{*}$ are related, then the mean-set attack can still work.

Finally, we want to mention again that probability measures on groups were not extensively studied and there are no good probability measures known on general groups and no general methods to construct measures satisfying the desired properties. Moreover, the problem of making distributions with large mean-sets is very complicated because not every subset of a group G can be realized as a mean-set. See [27] and [26] for more details. A number of open questions arise regarding the problems mentioned above, but dealing with them is beyond the scope of this paper.

7 Conclusion

In this paper, we used the probabilistic approach to analyze the Sibert et al. group-based authentication protocol. We have proved that the scheme does not meet necessary security compliances, i.e., it is not computationally zero-knowledge, in practice. To conduct our analysis, we introduced a new computational problem for finitely generated groups, the *shift search problem*, and employed probabilistic tools discussed in [27] to deal with the problem. In particular, the concept of the mean-set and the generalized strong law of large numbers for random group elements with values in the vertices of the connected and locally-finite Cayley graph of a given infinite finitely-generated group are used. The rate of success of getting the secret key, as a solution to the *shift search problem*, has been proved to be linear or exponential depending on the assumptions one is willing to make. In addition, we have provided experimental evidence that our approach is practical and can succeed even for braid groups. This work shows, among other things, that generalization of classical probabilistic results to combinatorial objects can lead to useful applications in group-based cryptography.

Bibliography

[1] I. Anshel, M. Anshel and D. Goldfeld, An algebraic method for public-key cryptography, *Math. Res. Lett.* **6** (1999), 287–291.

[2] J. Birman, *Braids, Links and Mapping Class Groups*, Annals of Math. Studies, Princeton University Press, 1974.

[3] J. S. Birman, V. Gebhardt and J. Gonzalez-Meneses, Conjugacy in Garside groups I: cyclings, powers, and rigidity, *Groups, Geometry, and Dynamics* **1** (2007), 221–279.

[4] J. S. Birman, V. Gebhardt and J. Gonzalez-Meneses, Conjugacy in Garside Groups III: Periodic braids, *J. Algebra* **316** (2007), 746–776.

[5] J. S. Birman, V. Gebhardt and J. Gonzalez-Meneses, Conjugacy in Garside groups II: Structure of the ultra summit set, *Groups, Geometry, and Dynamics* **2** (2008), 13–61.

[6] A. V. Borovik, A. G. Myasnikov and V. N. Remeslennikov, Multiplicative measures on free groups, *Int. J. Algebr. Comput.* **13** (2003), 705–731.

[7] A. Borovik, A. Myasnikov and V. Shpilrain, Measuring sets in infinite groups, in: *Computational and Statistical Group Theory*, Contemporary Mathematics 298, pp. 21–42, American Mathematical Society, 2002.

[8] R. Cramer and V. Shoup, A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack, in: *Advances in Cryptology – CRYPTO 1998*, Lecture Notes in Computer Science 1462, pp. 13–25, Springer-Verlag, London, UK, 1998.

[9] CRyptography And Groups (CRAG) C++ Library, available at `www.acc.stevens.edu/downloads.php`.

[10] P. Dehornoy, A fast method for comparing braids, *Adv. Math.* **125** (1997), 200–235.

[11] P. Dehornoy, Braid-based cryptography, in: *Group theory, statistics, and cryptography*, Contemporary Mathematics 360, pp. 5–33, American Mathematical Society, 2004.

[12] T. ElGamal, A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, *IEEE T. Inform. Theory* **IT-31** (1985), 469–473.

[13] D. B. A. Epstein, J. W. Cannon, D. F. Holt, S. V. F. Levy, M. S. Paterson and W. P. Thurston, *Word processing in groups*, Jones and Bartlett Publishers, 1992.

[14] U. Feige, A. Fiat and A. Shamir, Zero knowledge proofs of identity, *STOC '87: Proceedings of the nineteenth annual ACM Conference on Theory of Computing* (1987), 210–217.

[15] O. Goldreich, *Foundations of Cryptography: Volume 1, Basic Tools*, Cambridge University Press, 2001.

[16] O. Goldreich, *Zero-Knowledge twenty years after its invention*, preprint, available at `http://citeseer.ist.psu.edu/556429.html`, 2002.

[17] W. Hoeffding, Probability inequalities for sums of bounded random variables, *J. Am. Stat. Assoc.* **58** (1963), 13–30.

[18] I. Kapovich and A. G. Miasnikov, Stallings foldings and subgroups of free groups, *J. Algebra* **248** (2002), 608–668.

[19] C. Kessel and V. Turaev, *Barid Groups*, Graduate Texts in Mathematics, Springer, New York, 2009.

[20] K. H. Ko, S. J. Lee, J. H. Cheon, J. W. Han, J. Kang and C. Park, New public-key cryptosystem using braid groups, in: *Advances in Cryptology – CRYPTO 2000*, Lecture Notes in Computer Science 1880, pp. 166–183, Springer, Berlin, 2000.

[21] A. Kurosh, *Theory of Groups*, Chelsea Publishing Corp., 1979.

[22] J. Longrigg and A. Ushakov, Cryptanalysis of the shifted conjugacy authentication protocol, *J. Math. Crypt.* **2** (2008), 107–114.

[23] A. G. Miasnikov, Random Subgroups of Braid Groups: An Approach to Cryptanalysis of a Braid Group Based Cryptographic Protocol, in: *Advances in Cryptology – PKC 2006*, Lecture Notes in Computer Science 3958, pp. 302–314, Springer, Berlin, 2006.

[24] A. G. Miasnikov, *Group-based Cryptography*, Advanced Courses in Mathematics - CRM Barcelona, Birkhäuser Basel, 2008.

[25] A. G. Miasnikov, V. Shpilrain and A. Ushakov, A practical attack on some braid group based cryptographic protocols, in: *Advances in Cryptology – CRYPTO 2005*, Lecture Notes in Computer Science 3621, pp. 86–96, Springer, Berlin, 2005.

[26] N. Mosina, *Probability on graphs and groups: theory and applications*, Ph.D. thesis, Columbia University, 2009. Available at `www.math.columbia.edu/∼thaddeus/theses/2009/mosina.pdf`.

[27] N. Mosina and A. Ushakov, *Strong law of large numbers on graphs and groups*, available at `http://arxiv.org/abs/0904.1005`.

[28] N. Mosina and A. Ushakov, *Strong law of large numbers for metric spaces: central order*, in preparation.

[29] A. D. Myasnikov and A. Ushakov, Length Based Attack and Braid Groups: Cryptanalysis of Anshel-Anshel-Goldfeld Key Exchange Protocol, in: *Advances in Cryptology – PKC 2007*, Lecture Notes in Computer Science 4450, pp. 76–88, Springer, Berlin, 2007.

[30] A. D. Myasnikov and A. Ushakov, Cryptanalysis of Anshel-Anshel-Goldfeld-Lemieux key agreement protocol, *Groups-Complexity-Cryptology* **1** (2008), 263–75.

[31] M. Paterson and A. Razborov, The set of minimal braids is co-NP-complete, *J. Algorithms* **12** (1991), 393–408.

[32] G. Poupard and J. Stern, Security Analysis of a Practical "on the fly" Authentication and Signature Generation, in: *Advances in Cryptology – EUROCRYPT 1998*, Lecture Notes in Computer Science 1403, pp. 422–436, Springer, Berlin, 1998.

[33] C. P. Schnorr, Efficient identification and signatures for smart cards, in: *Advances in Cryptology – CRYPTO 1989*, Lecture Notes in Computer Science 435, pp. 239–252, Springer, Berlin, 1990.

[34] H. Sibert, P. Dehornoy and M. Girault, Entity authentication schemes using braid word reduction, *Discrete Appl. Math.* **154** (2006), 420–436.

[35] D. West, *Introduction to Graph Theory (2nd edition)*, Prentice Hall, 2000.

Received 18 June, 2009; revised 21 May, 2010.

Author information

Natalia Mosina, The City University of New York (LAGCC), Long Island City, NY, 11101, USA.
E-mail: nmosina@lagcc.cuny.edu

Alexander Ushakov, Department of Mathematics, Stevens Institute of Technology, Hoboken, NJ, 07030, USA.
E-mail: aushakov@stevens.edu