Chapter

The MOR Cryptosystem in Classical Groups with a Gaussian Elimination Algorithm for Symplectic and Orthogonal Groups

Sushil Bhunia, Ayan Mahalanobis, Pralhad Shinde and Anupam Singh

Abstract

In this chapter, we study the MOR cryptosystem with symplectic and orthogonal groups over finite fields of odd characteristics. There are four infinite families of finite classical Chevalley groups. These are special linear groups $\mathrm{SL}(d, q)$, orthogonal groups $\mathrm{O}(d, q)$, and symplectic groups $\mathrm{Sp}(d, q)$. The family $\mathrm{O}(d, q)$ splits into two different families of Chevalley groups depending on the parity of d. The MOR cryptosystem over $\mathrm{SL}(d, q)$ was studied by the second author. In that case, the hardness of the MOR cryptosystem was found to be equivalent to the discrete logarithm problem in $\mathbb{F}_{q^d}$. In this chapter, we show that the MOR cryptosystem over $\mathrm{Sp}(d, q)$ has the security of the discrete logarithm problem in $\mathbb{F}_{q^d}$. However, it seems likely that the security of the MOR cryptosystem for the family of orthogonal groups is $\mathbb{F}_{q^{d^2}}$. We also develop an analog of row-column operations in symplectic and orthogonal groups which is of independent interest as an appendix.

Keywords: public-key cryptography, MOR cryptosystem, Chevalley groups, Gaussian elimination, 2010 Mathematics Subject Classification: 94A60, 20H30

1. Introduction

Public-key cryptography is the backbone of this modern society. However with **recent advances in quantum computers** and its possible implication to factoring integers and solving the discrete logarithm problems, it seems that we are left with no secure cryptographic primitive. So it seems prudent that we set out in search for new cryptographic primitives and subsequently new cryptosystems. The obvious question is: how to search and where to look? One can look into several well-known hard problems in Mathematics and hope to create a trap-door function, or one can try to generalize the known, trusted cryptosystems.

This chapter is in the direction of generalizing a known cryptosystem with the hope that something practical and useful will come out of this generalization. A new but arbitrary cryptosystem might not be considered by the community as a secure cryptosystem for decades. So our approach is conservative but practical. Several such approaches were earlier made by many eminent mathematicians. To name a few, Maze et al. [1, 2] developed SAP and Shpilrain and Zapata developed CAKE, both work in non-abelian structures. There is an interesting cryptosystem in the work of Climent et al. [3]. We further recommend the work of Grogoriev et al. [4] and Roman'kov [5].

The cryptosystem that we have in mind is *the MOR cryptosystem* [6–9]. In Section 2, we describe the MOR cryptosystem in details. It is a simple but powerful generalization of the well-known and classic ElGamal cryptosystem. In this cryptosystem, the discrete logarithm problem works in the automorphism group of a group instead of the group. As a matter of fact, it can work in the automorphism group of most algebraic structures. However, we will limit ourselves to finite groups. One way to look at the MOR cryptosystem is that it generalizes the discrete logarithm problem from a cyclic (sub)group to an arbitrary group.

The MOR cryptosystem over $\mathrm{SL}(d, q)$ was studied earlier [6] and cryptanalyzed by Monico [10]. It became clear that working with matrix groups of size d over $\mathbb{F}_q$ and with automorphisms that act by conjugation, like the inner automorphisms, there are two possible reductions of the security to finite fields. It is the security of the discrete logarithm problem in $\mathbb{F}_{q^d}$ or $\mathbb{F}_{q^{d^2}}$ ([6], Section 7). This reduction is similar to the embedding of the discrete logarithm problem in the group of rational points of an elliptic curve to a finite field; the degree of the extension of that field over the field of definition of the elliptic curve is called the *embedding degree*. In the case of $\mathrm{SL}(d, q)$, it became the security of $\mathbb{F}_{q^d}$. The reason that we undertook this study is to see if the security in other classical Chevalley groups is $\mathbb{F}_{q^d}$ or $\mathbb{F}_{q^{d^2}}$.

In cryptography, it is often hard to come up with theorems about security of a cryptosystem. However, at this moment it seems likely that the security of the MOR cryptosystem in orthogonal groups $\mathrm{O}(d, q)$ is $\mathbb{F}_{q^{d^2}}$. The way we implement this cryptosystem is by solving the word problem in generators. It presents **no advantage** to small characteristic. In the light of Joux's [11] improvement of the index-calculus attack in small characteristic, this contribution of the MOR cryptosystem is remarkable.

In summary, the proposed MOR cryptosystem is totally different from the known ElGamal cryptosystems from a functional point of view. Its implementation depends on Gaussian elimination and substitutions (substituting a matrix for a word in generators). However, we do have a concrete and tangible understanding of its security. It is clear from this work that the MOR cryptosystem over classical groups is *not quantum-secure*. However, for other groups like solvable groups, the answer is not known and could be a topic of further research.

1.1 Structure of the chapter

This chapter is an interplay between computational group theory and public-key cryptography, in particular the MOR cryptosystem, and is thus interdisciplinary in nature. In this chapter, we study the MOR cryptosystem using the orthogonal and symplectic groups over finite fields of odd characteristic.

In Section 2, we describe the MOR cryptosystem in some details. We emphasize that the MOR cryptosystem is a natural generalization of the classic ElGamal cryptosystem. In Section 3, we describe the orthogonal and symplectic groups and their automorphisms. In Appendix A, we describe few new algorithms. These algorithms use row-column operations to write an element in classical groups as a word in generators. This is very similar to the Gaussian elimination algorithm for special linear groups. These algorithms are vital to the implementation of the MOR cryptosystem. These algorithms are also of *independent interest in computational group theory*.

1.2 Notations and terminology

It was bit hard for us to pick notations for this chapter. The notations used by a Lie group theorist is somewhat different from that of a computational group theorist. We tried to preserve the essence of notations as much as possible. For example, a Lie group theorist will use $\mathrm{SL}_{l+1}(q)$ to denote what we will denote by $\mathrm{SL}(l+1, q)$ or $\mathrm{SL}(d, q)$. We have used ${}^{T}X$ to denote the transpose of the matrix X. This was necessary to avoid any confusion that might arise when using X^{-1} and ${}^{T}X$ simultaneously. In this chapter, we use $\mathcal{K}$ and $\mathbb{F}_q$ interchangeably, while each of them is **a finite field of odd characteristic**. However, in the appendix the field k is unrestricted. The matrix te_{ij} is used to denote the matrix unit with t in the $(i,j)^{\mathrm{th}}$ place and zero everywhere else. We will often use $x_r(t)$ as generators, a notation used in the theory of Chevalley groups. Here r is a short hand for (i,j) and $x_r(t)$ are defined in **Tables A1, A3, A5,** and **A7**. We often refer to the orthogonal group as $\mathrm{O}(d, q)$, specifically, the split orthogonal group as $\mathrm{O}^{+}(2l, q)$ or $\mathrm{O}^{+}(2l+1, q)$ and the twisted orthogonal group as $\mathrm{O}^{-}(2l, q)$. All other notations used are standard.

2. The MOR cryptosystem

The MOR cryptosystem is a natural generalization of the classic ElGamal cryptosystem. It was first proposed by Paeng et al. [9]. To elaborate the idea behind a MOR cryptosystem, we take a slightly expository route. For the purpose of this exposition, we define **the discrete logarithm problem**. It is one of the most common cryptographic primitive in use. It works in any cyclic (sub)group $G = \langle g \rangle$ but is not secure in any cyclic group.

Definition 2.1 (The discrete logarithm problem). *The discrete logarithm problem in $G = \langle g \rangle$, given g and g^{m}, find* m.

The word "find" in the above definition is bit vague, in this chapter we mean compute m. The hardness to solve the discrete logarithm problem depends on the presentation of the group and is not an invariant under isomorphism. It is believed that the discrete logarithm problem is secure in the multiplicative group of a finite field and the group of rational points of an elliptic curve.

A more important cryptographic primitive, related to the discrete logarithm problem, is the **Diffie-Hellman problem**, also known as the **computational Diffie-Hellman problem**.

Definition 2.2 (Diffie-Hellman problem). *Given g, g^{m_1}, and g^{m_2}, find $g^{\mathrm{m}_1\mathrm{m}_2}$.*

It is clear; if one solves the discrete logarithm problem, then the Diffie-Hellman problem is solved as well. The other direction is not known.

The most prolific cryptosystem in use today is the ElGamal cryptosystem. It uses the cyclic group $G = \langle g \rangle$. It is defined as follows:

2.1 The ElGamal cryptosystem

A cyclic group $G = \langle g \rangle$ is public.

- **Public-key**: Let g and g^{m} be public.

- **Private-key**: The integer m be private.

Encryption:

To encrypt a plaintext $\mathfrak{M} \in G$, get an arbitrary integer $\mathrm{r} \in [1, |G|]$ and compute g^{r} and g^{rm}. The ciphertext is $(g^{\mathrm{r}}, \mathfrak{M} g^{\mathrm{rm}})$.

Decryption:

After receiving the ciphertext $(g^{\mathrm{r}}, \mathfrak{M} g^{\mathrm{rm}})$, the user uses the private-key m. So she computes g^{mr} from g^{r} and then computes $\mathfrak{M}$.

It is well known that the hardness of the ElGamal cryptosystem is equivalent to the Diffie-Hellman problem ([12], Proposition 2.10).

2.2 The MOR cryptosystem

In the case of the MOR cryptosystem, one works with the automorphism group of a group. An automorphism group can be defined on any algebraic structure, and subsequently a MOR cryptosystem can also be defined on that automorphism group; however, in this chapter we restrict ourselves to finite groups. Furthermore, we look at *classical groups* defined by generators and automorphisms that are defined as actions on those generators.

Let $G = \langle g_1, g_2, \ldots, g_s \rangle$ be a finite group. Let ϕ be a non-identity automorphism.

- Public-key: Let $\{\phi(g_i)\}_{i=1}^{s}$ and $\{\phi^{\mathrm{m}}(g_i)\}_{i=1}^{s}$ be public.
- Private-key: The integer m is private.

Encryption:

To encrypt a plaintext $\mathfrak{M} \in G$, get an arbitrary integer $\mathrm{r} \in [1, |\phi|]$ and compute ϕ^{r} and ϕ^{rm}. The ciphertext is $(\phi^{\mathrm{r}}, \phi^{\mathrm{rm}}(\mathfrak{M}))$.

Decryption:

After receiving the ciphertext $(\phi^{\mathrm{r}}, \phi^{\mathrm{rm}}(\mathfrak{M}))$, the user knows the private-key m. So she computes ϕ^{mr} from ϕ^{r} and then computes $\mathfrak{M}$.

Theorem 2.1 *The hardness to break the above MOR cryptosystem is equivalent to the Diffie-Hellman problem in the group $\langle \phi \rangle$.*

Proof. It is easy to see that if one can break the Diffie-Hellman problem, then one can compute ϕ^{mr} from ϕ^{m} in the public-key and ϕ^{r} in the ciphertext. This breaks the system.

On the other hand, observe that the plaintext is $\phi^{-\mathrm{mr}}(\phi^{\mathrm{mr}}(\mathfrak{M}))$. Assume that there is an oracle that can break the MOR cryptosystem, i.e., given ϕ, ϕ^{m} and a plaintext (ϕ^{r}, g) will deliver $\phi^{-\mathrm{mr}}(g)$. Now we query the oracle s times with the public-key and the ciphertext (ϕ^{r}, g_i) for $i = 1, 2, \ldots, s$. From the output, one can easily find $\phi^{\mathrm{mr}}(g_i)$ for $i = 1, 2, \ldots, s$. So we just witnessed that for ϕ^{m} and ϕ^{r}, one can compute ϕ^{mr} using the oracle. This solves the Diffie-Hellman problem.

In a practical implementation of a MOR cryptosystem, there are two things that matter the most.

a: The number of generators. As we saw that the automorphism ϕ is presented as action on generators. Larger the number of generators, bigger is the size of the public key.

b: Efficient algorithm to solve the word problem. This means that given $G = \langle g_1, g_2, \ldots, g_s \rangle$ and $g \in G$, is there an efficient algorithm to write g as word in $g_1, g_2, \ldots, g_s$? The reason of this importance is immediate—the automorphisms are presented as action on generators, and if one has to compute $\phi(g)$, then the word problem must be solved.

The obvious question is: what are the right groups for the MOR cryptosystem? In this chapter, we pursue a study of the MOR cryptosystem using **finite Chevalley groups** of classical type, in particular, orthogonal and symplectic groups.

3. Description of automorphisms of classical groups

This chapter studies the MOR cryptosystem for orthogonal and symplectic groups over a field of odd characteristics. As we discussed before, MOR cryptosystem is presented as action on generators of the group. Then to use an automorphism on an arbitrary element, one has to solve the word problem in that group with respect to that set of generators.

The generators and the Gaussian elimination algorithm to solve the word problem are described in Appendix A. We will be very brief here.

Let V be a vector space of dimension d over a field $\mathcal{K}$ of odd characteristic. Let $\beta : V \times V \to \mathcal{K}$ be a bilinear form. By fixing a basis of V, we can associate a matrix to β. We shall abuse the notation slightly and denote the matrix of the bilinear form by β itself. Thus $\beta(x,y)={}^{T}x\beta y$, where x, y are column vectors. We will work with non-degenerate bilinear forms and that means $\det\beta \neq 0$. A symmetric or skew-symmetric bilinear form β satisfies $\beta={}^{T}\beta$ or $\beta = -{}^{T}\beta$, respectively.

Definition 3.1 (Orthogonal group). *A square matrix X of size d is called orthogonal if ${}^{T}X\beta X = \beta$, where β is symmetric. It is well known that the orthogonal matrices form a group known as the orthogonal group.*

Definition 3.2 (Symplectic group). *A square matrix X of size d is called symplectic if ${}^{T}X\beta X = \beta$, where β is skew-symmetric. And the set of symplectic matrices form a symplectic group.*

We write the dimension of V as $d = 2l + 1$ or $d = 2l$ for $l \geq 1$. We fix a basis and index it by $0, 1, \ldots, l, -1, \ldots, -l$ in the odd dimension, and in the case of even dimension where there are two non-degenerate symmetric bilinear forms up to equivalence, we index the bases by $1, 2, \ldots, l, -1, -2, \ldots, -l$ and $1, -1, 2, \ldots, l, -2, \ldots, -l$ for split and twisted forms, respectively. We consider the non-degenerate bilinear forms β on V given by the following matrices:

a: The odd-orthogonal group. The form β is symmetric with $d = 2l + 1$ and

$$\beta = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 0 & I_l \\ 0 & I_l & 0 \end{pmatrix}.$$

b: The symplectic group. The form β is skew-symmetric with $d = 2l$ and

$$\beta = \begin{pmatrix} 0 & I_l \\ -I_l & 0 \end{pmatrix}.$$

c: The split orthogonal group. The form β is symmetric with $d = 2l$ and

$$\beta = \begin{pmatrix} 0 & I_l \\ I_l & 0 \end{pmatrix}.$$

c′: The twisted orthogonal group. The form β is symmetric with $d = 2l$ and

$$\beta = \begin{pmatrix} \beta_0 & 0 & 0 \\ 0 & 0 & I_{l-1} \\ 0 & I_{l-1} & 0 \end{pmatrix},$$

where I_l is the identity matrix of size l over $\mathcal{K}$ and for a fixed non-square $\epsilon \in \mathcal{K}$,

$$\beta_0 = \begin{pmatrix} 1 & 0 \\ 0 & \epsilon \end{pmatrix}.$$

We now describe the automorphism group of the orthogonal and symplectic groups. This helps us in picking the right set of automorphisms for the MOR cryptosystem.

Definition 3.3 (Orthogonal similitude group). *The orthogonal similitude group is defined as the set of matrices X of size d as*

$$\mathrm{GO}(d,q) = \left\{X \in \mathrm{GL}(d,q) | {}^{T}X\beta X = \mu\beta, \mu \in \mathbb{F}_q^{\times}\right\},$$

where $d = 2l + 1$ or $2l$ and β is of type a, c, or c′, respectively.

Definition 3.4 (Symplectic similitude group). *The symplectic similitude group is defined as*

$$\mathrm{GSp}(2l,q) = \left\{X \in \mathrm{GL}(2l,q) | {}^{T}X\beta X = \mu\beta, \mu \in \mathbb{F}_q^{\times}\right\},$$

where β is of type b.

Here μ depends on the matrix X and is called the similitude factor. The similitude factor μ defines a group homomorphism from the similitude group to $\mathbb{F}_q^{\times}$, and the kernel is the orthogonal group $\mathrm{O}(d,q)$ when β is symmetric and symplectic group $\mathrm{Sp}(2l,q)$ and when β is skew-symmetric, respectively ([13], Section 12). Note that scalar matrices λI for $\lambda \in \mathbb{F}_q^{\times}$ belong to the center of similitude groups. The similitude groups are analog of what $\mathrm{GL}(d,q)$ is for $\mathrm{SL}(d,q)$. For a discussion of the diagonal automorphisms of Chevalley groups, we need the diagonal subgroups of the similitude groups.

Definition 3.5 (Diagonal group). *The diagonal groups are defined to be the group of non-singular diagonal matrices in the corresponding similitude group and are as follows: in the case of $GO(2l+1,q)$, it is*

$$\left\{\mathrm{diag}\left(\alpha, \lambda_1, \ldots, \lambda_l, \mu\lambda_1^{-1}, \ldots, \mu\lambda_l^{-1}\right) | \lambda_1, \ldots, \lambda_l, \alpha^2 = \mu \in \mathbb{F}_q^{\times}\right\},$$

and in the case of $\mathrm{GO}(2l,q)$ and $\mathrm{GSp}(2l,q)$, it is

$$\left\{\mathrm{diag}\left(\lambda_1, \ldots, \lambda_l, \mu\lambda_1^{-1}, \ldots, \mu\lambda_l^{-1}\right) | \lambda_1, \ldots, \lambda_l, \mu \in \mathbb{F}_q^{\times}\right\}.$$

Conjugation by these diagonal elements produces diagonal automorphisms in the respective Chevalley groups. To build a MOR cryptosystem, we need to work with the automorphism group of Chevalley groups. In this section we describe the automorphism group of classical groups following Dieudonne [14].

Conjugation automorphisms: If N is a normal subgroup of a group G, then the conjugation maps $n \mapsto gng^{-1}$ for $n \in N$ and $g \in G$ are called conjugation automorphisms of G. In particular, both inner automorphisms and diagonal automorphisms are examples of conjugation automorphisms.

Central automorphisms: Let $\chi : G \to \mathcal{Z}(G)$ be a homomorphism to the center of the group. Then the map $g \mapsto \chi(g)g$ is an automorphism of G, known as the central automorphism. There are no nontrivial central automorphisms for perfect groups, for example, the Chevalley groups $\mathrm{SL}(l+1,\mathcal{K})$ and $\mathrm{Sp}(2l,\mathcal{K})$, $|\mathcal{K}| \geq 4$, and $l \geq 2$. In the case of orthogonal group, the center is of two elements $\{\mathrm{I}, -\mathrm{I}\}$, where I is the identity matrix. This implies that there are at most four central automorphisms in this case.

Field automorphisms: Let $f \in \mathrm{Aut}(\mathcal{K})$. In terms of matrices, field automorphisms amount to replacing each term of the matrix by its image under f.

Graph automorphisms: A symmetry of Dynkin diagram induces such automorphisms. This way we get automorphisms of order 2 for $\mathrm{SL}(l+1,\mathcal{K})$ and $l \geq 2$ and $\mathrm{O}^{+}(2l,\mathcal{K})$ and $l \geq 4$. We also get an automorphisms of order 3 for $\mathrm{O}^{+}(4,\mathcal{K})$.

In the case of $\mathrm{SL}(d, q)$ for $d \geq 3$, the map $x \mapsto A^{-1T}x^{-1}A$, where

$$A = \begin{pmatrix} 0 & \cdots & 0 & 0 & 0 & 1 \\ 0 & \cdots & 0 & 0 & -1 & 0 \\ 0 & \cdots & 0 & 1 & 0 & 0 \\ 0 & \cdots & -1 & 0 & 0 & 0 \\ \vdots & \iddots & \vdots & \vdots & \vdots & \vdots \\ (-1)^{l-1} & \cdots & 0 & 0 & 0 & 0 \end{pmatrix}$$

explicitly describes the graph automorphism.

In the case of $\mathrm{O}(2l, q)$ for $l \geq 5$, the graph automorphism is given by $x \mapsto B^{-1}xB$ where B is a permutation matrix obtained from identity matrix of size $2l \times 2l$ by switching the l^{th} row and $-l^{\text{th}}$ row. This automorphism is a conjugating automorphism.

Theorem 3.1 (Dieudonne). *Let $\mathcal{K}$ be a field of odd characteristic and $l \geq 2$.*

1. For the group $\mathrm{SL}(l+1,\mathcal{K})$, any automorphism is of the form $\iota\gamma\theta$ where ι is a conjugation automorphism defined by elements of $\mathrm{GL}(l+1,\mathcal{K})$ and γ is a graph automorphism for the special linear group.

2. For the group $\mathrm{O}^{+}(d,\mathcal{K})$, any automorphism is of the form $c_\chi\iota\theta$ where c_χ is a central automorphism and ι is a conjugation automorphism by elements of $\mathrm{GO}^{+}(d,\mathcal{K})$ (this includes the graph automorphism of even-orthogonal groups).

3. For the group $\mathrm{O}^{-}(d,\mathcal{K})$, any automorphism is of the form $\iota\theta$, where ι is a conjugation automorphism by elements of $\mathrm{GO}^{-}(d,\mathcal{K})$.

4. For the group $\mathrm{Sp}(2l,\mathcal{K})$, any automorphism is of the form $\iota\theta$ where ι is a conjugation automorphism by elements of $\mathrm{GSp}(2l,\mathcal{K})$.

In all cases θ denotes a field automorphism.

For a proof of the above theorem, see [26], Theorems 30 and 36. In the above theorem, conjugation automorphisms are given by conjugation by elements of a larger group, and it includes the group of inner automorphisms. We introduce diagonal automorphisms to make it more precise. The conjugation automorphisms ι can be written as a product of ι_g and η where ι_g is an inner automorphism and η is a diagonal automorphism.

Diagonal automorphisms: In the definition of the conjugating automorphism, when the conjugating element is from the similitude group but not in the group we get a diagonal automorphism. In the case of special linear groups, diagonal automorphisms are given by conjugation by diagonal elements of $\mathrm{PGL}(l + 1, q)$ on $\mathrm{PGL}(l + 1, q)$. In the case of symplectic and orthogonal groups, diagonal automorphisms are given by conjugation by corresponding diagonal group elements defined in Definition 3.5.

4. Security of the proposed MOR cryptosystem

The purpose of this section is to show that for a secure MOR cryptosystem over the classical Chevalley and twisted orthogonal groups, we have to look at automorphisms that act by conjugation like the inner automorphisms. There are other automorphisms that also act by conjugation, like the diagonal automorphism and the graph automorphism for odd-order orthogonal groups. Then we argue what is the hardness of our security assumptions. We denote the split orthogonal group by $O^+(2l, q)$ and twisted orthogonal group by $O^-(2l, q)$. Now onwards $O(2l,q)$ means either split or twisted orthogonal group and we will specify whenever required.

Let ϕ be an automorphism of one of the classical Chevalley groups G: $SL(l+1, q)$, $O(2l+1, q)$, $Sp(2l, q)$, or $O(2l, q)$. From Theorem 3.1, we know that $\phi = c_\chi \iota \eta \gamma \theta$ where c_χ is a central automorphism, ι is an inner automorphism, η is a diagonal automorphism, γ is a graph automorphism, and θ is a field automorphism.

The group of central automorphisms are too small and the field automorphisms reduce to a discrete logarithm in the field $\mathbb{F}_q$. So there is no benefit of using these in a MOR cryptosystem. Also there are not many graph automorphisms in classical Chevalley and twisted orthogonal groups other than special linear groups and odd-order orthogonal groups. In the odd-order orthogonal groups, these automorphisms act by conjugation. Recall here that our automorphisms are presented as action on generators. It is clear ([6], Section 7) that if we can recover the conjugating matrix from the action on generators, the security is a discrete logarithm problem in $\mathbb{F}_{q^d}$, or else the security is a discrete logarithm problem in $\mathbb{F}_{q^{d^2}}$.

So from these we conclude that for a secure MOR cryptosystem, we must look at automorphisms that act by conjugation, like the inner automorphisms. Inner automorphisms form a normal subgroup of $\mathrm{Aut}(G)$ and usually constitute the bulk of automorphisms. If ϕ is an inner automorphism, say $\iota_g : x \mapsto gxg^{-1}$, we would like to determine the conjugating element g. For the special linear group, it was done in [6]. We will follow the steps there for the present situation too. However, before we do that, let us digress briefly to observe that $G \to \mathrm{Inn}(G)$ given by $g \mapsto \iota_g$ is a surjective group homomorphism. Thus if G is generated by $g_1, g_2, \ldots, g_s$, then $\mathrm{Inn}(G)$ is generated by $\iota_{g_1}, \ldots, \iota_{g_s}$. Let $\phi \in \mathrm{Inn}(G)$. If we can find $g_j, j \in \{1, 2, \ldots, s\}$ generators, such that $\phi = \prod_j \iota_{g_j}$, then $\phi = \iota_g$ where $g = \prod_j g_j$. This implies that our problem is equivalent to solving the word problem in $\mathrm{Inn}(G)$. Note that solving word problem depends on how the group is presented and it is not invariant under group homomorphisms. Thus the algorithm described earlier to solve the word problem in the classical Chevalley and twisted orthogonal groups does not help us in the present case.

In what follows, we will use generators $x_r(t)$, where $r = (i,j); \quad i \neq j, \quad 1 \leq i, j \leq d$ for the special linear group. For symplectic group $r = (i,j); i, j \in \{\pm 1, \pm 2, \ldots, \pm l\}$. For the even-orthogonal group, $r = (i,j); i,j \in \{\pm 1, \pm 2, \ldots, \pm l\}; \pm i \neq \pm j$. For the odd-orthogonal group $r = (i,j); -l \leq i \leq l$ and $j \in \{\pm 1, \pm 2, \ldots \pm l\}; \pm i \neq \pm j$. These are the Chevalley generators for the Chevalley groups we are dealing with and are described in details in **Tables A1, A5, A3, and A7** in the Appendix.

4.1 Reduction of security

In this subsection, we show that for special linear and symplectic groups, the security of the MOR cryptosystem is the hardness of the discrete logarithm problem in $\mathbb{F}_{q^d}$. This is the same as saying that we can find the conjugating matrix up to a scalar multiple. We further show that the method that works for special linear and symplectic groups does not work for orthogonal groups.

Let ϕ be an automorphism that works by conjugation, i.e., $\phi = \iota_g$, for some g, and we try to determine g.

Step 1: The automorphism ϕ is presented as action on generators $x_r(t)$.

Thus $\phi(x_r(t)) = g(I + te_r)g^{-1} = I + tge_rg^{-1}$. This implies that we know ge_rg^{-1} for all possible r. We first claim that we can determine $N = gD$ where D is sparse, in fact, diagonal in the case of special linear and symplectic groups.

In the case of special linear groups, write $g = [G_1, \ldots, G_i, \ldots, G_d]$, where G_i are column vectors of g. Then $ge_{i,j} = [G_1, \ldots, G_d]e_{i,j} = [0, \ldots, 0, G_i, 0\ldots, 0]$ where G_i is at the j^{th} place. Multiplying this with g^{-1} on the right, i.e., computing $ge_{i,j}g^{-1}$, determines G_i up to a scalar multiple d_i (say). Thus, we know $N = gD$ where $D = \text{diag}(d_1, \ldots, d_{l+1})$.

For the symplectic groups, we do the similar computation with the generators $I + te_{i,-i}$ and $I + te_{-i,i}$. Write g in the column form as $[G_1, \ldots G_l, G_{-1}, \ldots, G_{-l}]$. Now,

1. $[G_1, \ldots G_l, G_{-1}, \ldots, G_{-l}]e_{i,-i} = [0, \ldots, 0, G_i, 0, \ldots, 0]$ where G_i is at $-i$th place. Multiplying this further with g^{-1} gives us scalar multiple of G_i, say d_iG_i.

2. $[G_1, \ldots G_l, G_{-1}, \ldots, G_{-l}]e_{-i,i} = [0, \ldots, 0, G_{-i}, 0, \ldots, 0]$ where G_{-i} is at ith place. Multiplying this with g^{-1} gives us scalar multiple of G_{-i}, say $d_{-i}G_i$.

Thus we get $N = gD$ where D is a diagonal matrix $\text{diag}(d_1, \ldots, d_l, d_{-1}, \ldots, d_{-l})$.

Step 2: Compute $N^{-1}\phi(x_r(t))N = D^{-1}g^{-1}\left(gx_r(t)g^{-1}\right)gD = I + D^{-1}e_rD$ which is equivalent to computing $D^{-1}e_rD$.

In the case of special linear groups, we have D a diagonal. Thus by computing $D^{-1}e_{i,j}D$, we determine $d_i^{-1}d_j$ for $i \neq j$ and form a matrix $\text{diag}\left(1, d_2^{-1}d_1, \ldots, d_l^{-1}d_1\right)$, and multiplying this to N, we get d_1g. Hence we can determine g up to a scalar matrix.

For symplectic groups, we can do similar computation as D is diagonal. First compute $D^{-1}\left(e_{i,j} - e_{-j,-i}\right)D$ to get $d_i^{-1}d_j$ and $d_{-i}^{-1}d_{-j}$ for $i \neq j$. Now compute $D^{-1}e_{i,-i}D, D^{-1}e_{-i,i}D$ to get $d_id_{-i}^{-1}, d_{-i}d_i^{-1}$. We form a matrix

$$\text{diag}\left(1, d_2^{-1}d_1, \ldots, d_l^{-1}d_1, d_{-1}^{-1}d_{-2}.d_{-2}^{-1}d_2.d_2^{-1}d_1, \ldots, d_{-l}^{-1}d_{-1}.d_{-1}^{-1}d_1\right)$$

and multiply it to $N = gD$ to get d_1g. Thus we can determine g up to a scalar multiple say ag. Similarly we can determine g^{m} up to a scalar multiple say bg^{m}. Now, compute $(ag)^{q-1} = g^{q-1}$ and $(bg^{\text{m}})^{q-1} = (g^{\text{m}})^{q-1}$, and then we can recover m by solving the discrete logarithm in the matrices using Menezes and Wu's idea [15]. However, if we choose g such that $g^{q-1} = 1$, then it seems that we might avoid this line of attack. We can bypass this argument by recovering the scalars a and b, and then to determine m, we compute the discrete logarithm in $\langle g \rangle$ using Menezes and Wu's idea. We prove the following proposition.

Proposition 4.1 *Given any $g \in \text{Sp}(d, q)$ up to scalar multiple ag, $a \in \mathbb{F}_q$. If $\gcd(d, q-1) = 1$, we can determine the scalar a. Otherwise one can find the scalar a by solving a discrete logarithm problem in $\mathbb{F}_q$.*

Proof. We can recover the scalar a as follows: Let $\{\lambda_1, \ldots, \lambda_d\}$ be a set of eigenvalues of g, and then the eigenvalues of ag are $\{a\lambda_1, \ldots, a\lambda_d\}$. Set $\alpha = a\lambda_1 \cdots a\lambda_d$ and thus $\alpha = a^d$ as $\lambda_1 \cdots \lambda_d = det(g) = 1$. Suppose $\gcd(d, q-1) = \zeta$, using extended Euclidean algorithm, we find u and v such that $ud + v(q-1) = \zeta$. Next, computing α^u, we get $a^{ud} = a^{\zeta - v(q-1)} = a^{\zeta}$. Thus, if $\gcd(d, q-1) = 1$, then we have recovered the scalar a; otherwise we can recover the scalar by solving the discrete logarithm problem in $\mathbb{F}_q$.

Thus, if $\gcd(d, q-1) = 1$, then using the above proposition, we can recover the scalars a and b from ag and bg^{m}, respectively. Otherwise one needs to solve discrete logarithm problem in $\mathbb{F}_q$ to recover the scalars. Now, we can recover g and g^{m} from ag and bg^{m} just by multiplying with scalar matrices $a^{-1}I$ and $b^{-1}I$, respectively. Finally, we recover m using Menezes and Wu's idea. Thus, if we choose g such that

$g^{q-1} = 1$ and $\gcd(d, q-1) \neq 1$, then to solve the discrete logarithm in $\langle \phi \rangle$, one needs to solve the discrete logarithm in $\mathbb{F}_q$ and $\mathbb{F}_{q^d}$.

However, in the case of orthogonal groups, we show that one cannot recover g up to a diagonal matrix using the above approach, and hence the above reduction attack does not work.

Theorem 4.1 *Let $g \in \mathrm{GO}(d, q)$. Consider the conjugation automorphism $\phi : \mathrm{O}(d, q) \to \mathrm{O}(d, q)$. Let $\{x_r\}$ be a set of Chevalley generators of $\mathrm{O}(d,q)$ described in Appendix A. Suppose that the public-key is presented as an action of ϕ on $\{x_r\}$, then it is impossible to recover a matrix gD, where D is a diagonal matrix using the above reduction.*

Proof. We prove the theorem for $\mathrm{O}^{+}(d, q)$, d even, and the theorem follows for other cases similarly. Let $d = 2l$ and we write g in columns form as $g = [C_1, \ldots, C_l, C_{-1}, \ldots, C_{-l}]$. We compute $ge_r g^{-1}$ which gives the following equations:

1. Note that $g(e_{i,j} - e_{-j,-i})g^{-1} = [0, \ldots, 0, C_i, 0, \ldots, 0, C_{-j}, 0, \ldots, 0]g^{-1}$, where C_i is at jth place and C_{-j} is at $-i^{\text{th}}$ place. After multiplying by g^{-1}, we get a matrix whose all columns are linear combinations of columns C_i and C_{-j}.

2. Note that $g(e_{i,-j} - e_{j,-i})g^{-1} = [0, \ldots, 0, C_i, 0, \ldots, 0, C_j, 0, \ldots, 0]g^{-1}$, where C_i is at $-j^{\text{th}}$ place and C_j is at $-i^{\text{th}}$ place. After multiplying by g^{-1}, we get a matrix whose all columns are linear combinations of columns C_i and C_j.

3. Note that $g(e_{-i,j} - e_{-j,i})g^{-1} = [0, \ldots, 0, C_{-i}, 0, \ldots, 0, C_{-j}, 0, \ldots, 0]g^{-1}$, where C_{-i} is at j^{th} place and C_{-j} is at i^{th} place. After multiplying by g^{-1}, we get a matrix whose all columns are linear combinations of columns C_{-i} and C_{-j}.

Suppose one can construct a matrix B from columns obtained above such that $B = gD$, where D is diagonal, then we can see that $d_i C_i = a_i C_j + b_j C_k$ for some i, j, k which is a contradiction as $\det(g) \neq 0$. Thus, it is not possible to construct a matrix B such that $B = gD$, where D is diagonal.

This conclusively proves that the attack on the special linear groups and symplectic groups will not work for most orthogonal groups.

For orthogonal groups, the best we can do is the following: We can construct N such that $N = g(D_1 + PD_2)$, where D_1 and D_2 are diagonal and P is a permutation matrix. We demonstrate the construction of N in the case of a split orthogonal group $\mathrm{O}^{+}(2l, q)$; similar construction works for other cases as well. Computing $ge_r g^{-1}$ gives the following equations:

1. $[G_1, \ldots G_l, G_{-1}, \ldots, G_{-l}](e_{i,j} - e_{-j,-i})g^{-1} = [0, \ldots, 0, G_i, 0, \ldots, 0, G_{-j}, 0, \ldots, 0]g^{-1}$, where G_i is at jth place and G_{-j} is at $-i^{\text{th}}$ place. This gives us a linear combination of the columns G_i and G_{-j}.

2. $[G_1, \ldots G_l, G_{-1}, \ldots, G_{-l}](e_{i,-j} - e_{j,-i})g^{-1} = [0, \ldots, 0, G_i, 0, \ldots, 0, G_j, 0, \ldots, 0]g^{-1}$, where G_i is at $-j^{\text{th}}$ place and G_j is at $-i^{\text{th}}$ place. This will give us a linear combination of the columns G_i and G_j.

3. $[G_1, \ldots G_l, G_{-1}, \ldots, G_{-l}](e_{-i,j} - e_{-j,i})g^{-1} = [0, \ldots, 0, G_{-i}, 0, \ldots, 0, G_{-j}, 0, \ldots, 0]g^{-1}$, where G_{-i} is at jth place and G_{-j} is at ith place. This will give us a linear combination of the columns G_{-i} and G_{-j}.

We construct a matrix N as follows: For each $i = 1, \ldots, l-1$, compute $g\left(I + e_{i,i+1} - e_{-(i+1),-i}\right)g^{-1} - I$ whose each column is a linear combination of C_i and $C_{-(i+1)}$. Choose one of its column say $r_iC_i + s_iC_{-(i+1)}$ for each $i = 1, \ldots, l-1$. Similarly compute $g\left(I + e_{i+1,i} - e_{-i,-(i+1)}\right)g^{-1} - I$ and choose $r_{-i}C_{-i} + s_{-i}C_{(i+1)}$ for each $i = 1, \ldots, l-1$. Further, we compute $g(I + e_{1,-l} - e_{l,-1})g^{-1} - I$ to get $r_lC_l + s_lC_1$ and $g(I + e_{-1,l} - e_{-l,1})g^{-1} - I$ to get $r_{-l}C_{-l} + s_{-l}C_{-1}$. We set $N = [r_1C_1 + s_1C_{-2}, \ldots, r_{l-1}C_{l-1} + s_{l-1}C_{-l}, r_lC_l + s_lC_1, r_{-1}C_{-1} + s_{-1}C_2, \ldots, r_{-(l-1)}C_{-(l-1)} + s_{-(l-1)}C_l, r_{-l}C_{-l} + s_{-l}C_{-1}]$. Now it is easy to note that $N = g(D_1 + PD_2)$, where $D_1 = \text{diag}(r_1, \ldots, r_l, r_{-1}, \ldots, r_{-l})$, $D_2 = \text{diag}(s_1, \ldots, s_l, s_{-1}, \ldots, s_{-l})$, and P are permutation matrix corresponding to the permutation of indexing set $1 \to -2 \to 3 \to -4 \to \cdots \to l-1 \to -l \to -1 \to 2 \to -3 \to 4 \to \cdots \to -(l-1) \to l \to 1$.

Thus we get $N = g(D_1 + PD_2)$, where D_1 and D_2 are diagonal and P is a permutation matrix. This is not a diagonal matrix. One can do a similar computation for the odd-orthogonal group and twisted orthogonal group as well.

Remark 4.1 *An observant reader would ask the question: why does this attack works for the special linear and symplectic groups but not for orthogonal groups? The answer lies in a closer look at the generators (elementary matrices) for these groups.*

In the special linear groups, the generators are the elementary transvections of the form $I + te_{i,j}$ where $i \neq j$ and $t \in \mathbb{F}_q$. Then the attack goes on smoothly as we saw earlier. However, when we look at generators of the form $I + te_{i,j} - te_{-j,-i}$, where $t \in \mathbb{F}_q$ and $i \neq j$, conjugating by them, it gets us a linear sum of the ith and jth column, not scalar multiple of one particular column. This stops the attack from going forward. However in the symplectic groups, there are generators of the form $I + e_{i,-i}$ and $I + e_{-i,i}$ for $1 \leq i \leq l$. These generators make the attack possible for the symplectic groups. However there are no such generators for orthogonal groups, and so this attack turns out to be impossible for orthogonal groups.

5. The case for two-generators and prime fields

One serious objection against a MOR cryptosystem is the size of the key ([10], Section 7). The reason is that in a MOR cryptosystem, the automorphisms are presented as action on generators. Now the bigger the number of generators, the larger the key-size.

On the other hand, many of the finite simple groups can be generated by two elements. However, a set of generators is not enough. We must be able to compute the image of an arbitrary element. When the automorphism is presented as action on generators, we need an efficient solution to the word problem in order to do that. We have demonstrated in Appendix A that there is one set of generators, the elementary matrices, for which the word problem is easy.

The theme of this section is that for symplectic and even-order split orthogonal groups, there are two generators and for the odd-orthogonal group there are three generators. Over the **prime field of odd characteristic**, one can easily compute the word corresponding to the elementary matrices for these generators.

So one can present the automorphisms ϕ and ϕ^m as action on these few generators and then compute the action of these automorphisms on the elementary matrices later. This substantially reduces the key-size. To do this we use the technique of *straight line programs*, which is popular in computational group theory. These are programs, but in practice are actually easy to use formulas. Say, for example, we want to compute $x_{i,j}(t)$ for some $t \in \mathbb{F}_q$. We have loaded matrices $w^{i-1}x_{1,2}(\cdot)w^{(i-1)}$ in the memory in such a way that this formula takes as input t and put it in the (1, 2) position of the matrix $x_{1,2}(\cdot)$ and do the matrix multiplication. This is one straight line program. Since these programs are loaded in the memory, computation is much faster. This is somewhat similar to a time-memory trade-off. We have built a series of these straight line programs, where one straight line program can use other straight line programs and have written down the length of these programs. The length is nothing but the number of matrices in the formula.

Using the symplectic group in the MOR cryptosystem is straightforward. However, using orthogonal groups is little tricky because of the presence of λ in the output of the Gaussian elimination algorithm (see Section A.2.3). It is well known that the elementary matrices, without w_i—the row interchanges matrices and generates Ω, the commutator subgroup of a orthogonal group. However in between the commutator and the whole group, there is another important subgroup, $W\Omega = \langle \Omega, w_i \rangle$ for some i. From the algorithmic point of view, it is the subgroup of all the matrices for which the λ is a square. Now once the λ is a square and we can efficiently compute the square root, we can write this matrix down as product of elementary matrices, and it is easy to implement in the MOR cryptosystem. It is well known that if $p \equiv 3(\text{mod}4)$, then it is easy to compute the square root. Only for this reason, in the latter part of this section and for orthogonal groups, we concentrate on $p \equiv 3\ (\text{mod}4)$.

5.1 Symplectic group Sp $(2l, p)$

Let p be an odd prime. It is known [16] that the group Sp$(2l,p)$ is generated by two elements:

$$x = x_{1,2}(1) \tag{1}$$

$$w = \begin{pmatrix} 0 & 1 \\ -I_{2l-1} & 0 \end{pmatrix} \tag{2}$$

We will refer these two elements as **Steinberg generators**. However in the context of the MOR cryptosystem, we need to know how to go back and forth between these two generating sets—Steinberg generators and elementary matrices (see **Table A3**). To write w as a product of elementary matrices is easy, just put this generator through our Gaussian elimination algorithm. Here we demonstrate the other way round, that is, how to write elementary matrices as a product of x and w. In what follows, we denote the length of SLPs by $L(\delta, i)$, where $\delta = j - i$ and $1 \le i < j \le l$.

$$\begin{array}{lll}
\delta = 1, & x_{i,j}(t) = & w^{i-1}x_{1,2}(t)w^{-(i-1)}, \\
\delta = 2, & x_{i,j}(t) = & \left[x_{i,j-1}(t), x_{j-1,j}(1)\right], \\
\delta = 3, & x_{i,j}(t) = & \left[x_{i,j-1}(t), x_{j-1,j}(1)\right], \\
\vdots & \vdots & \vdots \\
\delta = l-1, & x_{i,j}(t) = & \left[x_{i,j-1}(t), x_{j-1,j}(1)\right].
\end{array}$$

Here

$$L(\delta, i) = \begin{cases} 2i - 1 & \text{for } \delta = 1, \\ 2L(\delta - 1) + 4(i + \delta) - 6 & \text{for } \delta = 2, 3, \ldots, l - 1. \end{cases}$$

Now $w^l = (-1)^{l-1}\begin{pmatrix} 0 & I_l \\ -I_l & 0 \end{pmatrix}$ and $x_{j,i}(t) = w^l x_{i,j}(-t) w^{-l}$, so length of this SLP is $L(\delta, i) + 2l$. Hence we get all $x_{i,j}(t)$ for $1 \le i \ne j \le l$. Number of SLP is l. Next observe the following:

Elements	Indices	Equation	Length	
$x_{1,-l}(t)$		$wx_{l-1,l}(t)w^{-1}$	$2l-1$	
$x_{1,-i}(t)$	$2 \le i \le l-1$	$[x_{i,l}(t), x_{1,-l}(1)]$	$2(L(l-i,i)+2l-1)$	
$x_{i,-j}(t)$	$2 \le i \le l-1$ $(i+1 \le j \le l)$	$[x_{i,1}(t), x_{1,-j}(1)]$	$2(L(i-1,1)+4l-1)$ $2(L(i-1,1)+2L(l-j,j)+6l-2)$	$j=l$ $j \ne l$
$x_{i,-i}(t)$	$i = 1, 2, \ldots, l-1$	$\left[x_{i,i+1}\left(\frac{t}{2}\right), x_{i,-(i+1)}(1)\right]$	$2(2L(l-2,1)+10l-5)$ $2(L(1,i)+2L(i-1,1)+$ $4L(l-(i+1),i+1)+12l-4)$	$i=l-1$ $i \ne l-1$
$x_{l,-l}(t)$		$\left[x_{l,l-1}\left(\frac{t}{2}\right), x_{l-1,-l}(1)\right]$	$2(2L(l-2,1)+12l-5)$	

So we generate all $x_{i,-j}(t)$ for $1 \le i < j \le l$ and $x_{i,-i}(t)$ for $1 \le i \le l$. Now $w^l x_{i,-j}(t) w^{-l} = x_{-i,j}(t)$ for $1 \le i < j \le l$ and $w^l x_{i,-i}(t) w^{-l} = x_{-i,i}(t)$ for $1 \le i \le l$, then we get $x_{-i,j}(t)$ and $x_{-i,i}(t)$. Total number of SLPs is $l + (3+1) + (2+1) = l + 7$. Hence we generate all the elementary matrices (**Table A3**) using only two generators x and w. Hence $\mathrm{Sp}(2l, p)$ is generated by only two generators x and w.

5.2 Split orthogonal group $\mathrm{O}^{+}(2l, p)$

Let $p \equiv 3 \pmod 4$ be a prime. It is known [16] that the group $\mathrm{O}^{+}(2l,p)$ is generated by two elements:

$$x = x_{1,2}(1), \tag{3}$$

$$w = \left(\begin{array}{ccc|ccc} 0 & \cdots & 0 & 0 & \cdots & 1 \\ -1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \cdots & -1 & 0 & 0 & \cdots & 0 \\ \hline 0 & \cdots & 1 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & -1 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & -1 & 0 \end{array}\right) \tag{4}$$

We will refer these two elements as **Steinberg generators**. As we discussed earlier, in context of the MOR cryptosystem, we need to know how to go back and forth between these two generating sets—Steinberg generators and elementary matrices (**Table A1**). To write w as a product of elementary matrices is easy, just put this generator through our Gaussian elimination algorithm. Here we demonstrate the other way round, that is, how to write elementary matrices as a product of x and w. In what follows, we denote the length of SLPs by $L(\delta, i)$, where $\delta = j - i$ and $1 \le i < j \le l$.

$$\begin{aligned}
\delta = 1, &\quad x_{i,j}(t) = && w^{i-1}x_{1,2}(t)w^{-(i-1)},\\
\delta = 2, &\quad x_{i,j}(t) = && \left[x_{i,j-1}(t), x_{j-1,j}(1)\right],\\
\delta = 3, &\quad x_{i,j}(t) = && \left[x_{i,j-1}(t), x_{j-1,j}(1)\right],\\
\vdots &\quad \vdots && \vdots\\
\delta = l-1, &\quad x_{i,j}(t) = && \left[x_{i,j-1}(t), x_{j-1,j}(1)\right].
\end{aligned}$$

Here

$$L(\delta, i) = \begin{cases} 2i-1 & \text{for } \delta = 1,\\ 2L(\delta-1) + 4(i+\delta) - 6 & \text{for } \delta = 2, 3, \dots, l-1. \end{cases}$$

Now $w^l = (-1)^l \begin{pmatrix} 0 & I_l \\ I_l & 0 \end{pmatrix}$ and $x_{j,i}(t) = w^l x_{i,j}(-t)w^{-l}$, so length of this SLP is $L(\delta, i) + 2l$. Hence we get all $x_{i,j}(t)$ for $1 \le i \ne j \le l$. The number of SLPs is l. Next observe the following:

Elements	Indices	Equation	Length	
$x_{1,-l}(t)$		$wx_{l-1,l}(t)w^{-1}$	$2l-1$	
$x_{1,-i}(t)$	$2 \le i \le l-1$	$[x_{i,l}(t), x_{1,-l}(1)]$	$2(L(l-i,i) + 2l - 1)$	
$x_{i,-j}(t)$	$2 \le i \le l-1$ $(i+1 \le j \le l)$	$\left[x_{i,1}(t), x_{1,-j}(1)\right]$	$2(L(i-1,1) + 2L(l-j,j) + 6l - 2)$ $2(L(i-1,1) + 4l - 1)$	$j \ne l$ $j = l$

So we generate all $x_{i,-j}(t)$ for $i > j$. Now $w^l x_{i,-j}(t)w^{-l} = x_{-i,j}(t)$, and we get $x_{-i,j}(t)$ and the total number of SLPs is $l + 4$. It is shown by Ree [17] that elementary matrices $x_{i,j}(t)$ generate $\Omega(2l, p)$, the commutator subgroup of $\mathrm{O}(2l, p)$. Hence we generate $\Omega(2l, p)$, using only two elements x and w. Since we generate $x_{i,j}(t)$ and $w_{i,j}$ as a product of $x_{i,j}(t)$ and $w = w_{1,2}(1)w_{2,3}(1)\cdots w_{l-1,l}(1)w_l$, so we are able to generate w_l. Here $w_{i,j}(t) = x_{i,j}(t)x_{j,i}(-t^{-1})x_{i,j}(t)$ for $i \ne j$ and $w_l = I - e_{l,l} - e_{-l,-l} + e_{l,-l} + e_{-l,l}$. Now we know $w_{l-1} = w_l w_{l,l-1}(1)w_{l-1,-l}(1)$, so we generate w_{l-1}. Hence by induction, we generate $w_i = w_{i+1}w_{i+1,i}(1)w_{i,-(i+1)}(1)$ for $i = l-1, \dots, 1$. Here $w_{i,-j}(t) = x_{i,-j}(t)(1)x_{-i,j}(t^{-1})x_{i,-j}(t)$, for $i < j$. Hence we generate all the elementary matrices (**Table A1**) using only two generators x and w. So we generate a new subgroup $W\Omega(2l, p)$ of $\mathrm{O}(2l,p)$, which is a normal subgroup of $\mathrm{O}(2l, p)$. Our algorithm output matrix is $d(\lambda) = \mathrm{diag}\left(1, 1, \dots, \lambda, 1, 1, \dots, \lambda^{-1}\right)$. If $\lambda \in F_p^{\times 2}$, say $\lambda \equiv t^2 (\mathrm{mod}\, p)$, then $t \equiv \lambda^{\frac{p+1}{4}} (\mathrm{mod}\, p)$, since $p \equiv 3 (\mathrm{mod}\ 4)$. Then

$$\begin{aligned}
d(\lambda) &= \mathrm{diag}\left(1, \dots, t^2, 1, \dots, , t^{-2}\right)\\
&= w_{l-1,l}(1)\mathrm{diag}\left(1, \dots, t^2, 1, 1, \dots, , t^{-2}, 1\right)w_{l-1,l}(-1)\\
&= w_{l-1,l}(1)w_{l-1,l}(t)w_{l-1,l}(-1)w_{l-1,-l}(t)w_{l-1,-l}(-1)w_{l-1,l}(-1).
\end{aligned}$$

Hence we generate $W\Omega(2l, p)$ using only two generators x and w.

5.3 Orthogonal group $\mathrm{O}(2l{+}1, p)$

Let $p \equiv 3$ (mod4) be a prime. It is known [16] that the group $\mathrm{O}(2l{+}1, p)$ is generated by these elements:

$$x = x_{0,1}(1), \tag{5}$$

$$w = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -I_{2l-1} & 0 \end{pmatrix}, \tag{6}$$

$$w_l = \quad I - e_{l,l} - e_{-l,-l} + e_{l,-l} + e_{-l,l}. \tag{7}$$

We will refer these three elements as **Steinberg generators**. However in context of the MOR cryptosystem, we need to know how to go back and forth between these two generating sets—Steinberg generators and elementary matrices (**Table A5**). To write w as a product of elementary matrices is easy, just put this generator through our Gaussian elimination algorithm. Here we demonstrate the other way round, that is, how to write elementary matrices as a product of w and x. First we compute, $x_{0,i}(t) = w^{i-1}x_{0,1}(1)w^{-(i-1)}$ which is of length $2i - 1$ for $1 \le i \le l$. Now

$$w^l = (-1)^l \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & I_l \\ 0 & I_l & 0 \end{pmatrix}$$

and $x_{i,0}(t) = w^l x_{0,i}(-t)w^{-l}$ for $1 \le i \le l$, and length of this SLP is $2l + 2i - 1$. So we get $x_{i,0}(t)$ and $x_{0,i}(t)$ for $i = 1, 2, \ldots, l$. Again we have $x_{1,2}(t) = \left[x_{1,0}\left(\frac{t}{2}\right), x_{0,2}(1)\right]$ and length of this SLP is $4l + 8$. In what follows, we denote the length of SLPs by $L(\delta, i)$, where $\delta = j - i$ and $1 \le i < j \le l$.

$$\begin{array}{rcl} \delta = 1, & x_{i,j}(t) = & w^{i-1}x_{1,2}(t)w^{-(i-1)}, \\ \delta = 2, & x_{i,j}(t) = & \left[x_{i,j-1}(t), x_{j-1,j}(1)\right], \\ \delta = 3, & x_{i,j}(t) = & \left[x_{i,j-1}(t), x_{j-1,j}(1)\right], \\ \vdots & \vdots & \vdots \\ \delta = l-1, & x_{i,j}(t) = & \left[x_{i,j-1}(t), x_{j-1,j}(1)\right]. \end{array}$$

Here

$$L(\delta, i) = \begin{cases} 2i + 4l + 6 & \text{for } \delta = 1, \\ 2L(\delta - 1, i) + 4(i + \delta + 2l + 2) & \text{for } \delta = 2, 3, \ldots, l - 1. \end{cases}$$

As $x_{j,i}(t) = w^l x_{i,j}(-t)w^{-l}$, so the length of this SLP is $L(\delta, i) + 2l$. Hence we generate all $x_{i,j}(t)$ for $1 \le i \ne j \le l$ and the number of SLPs is $3 + (l - 1) + 1 = l + 3$. Next observe the following:

Elements	Indices	Equation (SLP)	Length	
$x_{1,-l}(t)$		$wx_{l-1,l}(t)w^{-1}$	$6l + 6$	
$x_{1,-i}(t)$	$2 \le i \le l - 1$	$[x_{i,l}(t), x_{1,-l}(1)]$	$24l + 20$	$i = l - 1$
			$2L(l - i, i) + 12(l + 1)$	$i \ne l - 1$
$x_{i,-j}(t)$	$2 \le i \le l - 1$	$\left[x_{i,1}(t), x_{1,-j}(1)\right]$	$2L(i - 1, 1) + 4L(l - j - \delta, j - \delta) + 4(7l + 6)$	$j < l - 1$
	$(i + 1 \le j \le l)$		$2L(i - 1, 1) + 4(7l + 5)$	$j = l - 1$
			$2L(i - 1, 1) + 10l + 6$	$j = l$

So we generate all $x_{i,-j}(t)$ for $i<j$. Now $w^l x_{i,-j}(t)w^{-l} = x_{-i,j}(t)$, and we have $x_{-i,j}(t)$. The total number of SLPs is $l+7$. It is shown in Ree [17] that elementary matrices $x_{i,j}(t)$ generate $\Omega(2l+1,p)$, the commutator subgroup of $\mathrm{O}(2l+1,p)$ which is of index 4. So we generate $\Omega(2l+1,p)$, using only two generators x and w. Now we know $w_{l-1} = w_l w_{l,l-1}(1)w_{l-1,-l}(1)$, so we generate w_{l-1}. Hence inductively we can generate $w_i = w_{i+1}w_{i+1,i}(1)w_{i,-(i+1)}(1)$ for $i = l-1, \ldots, 1$. Here $w_{i,j}(t) = x_{i,j}(t)x_{j,i}(-t^{-1})x_{i,j}(t)$ for $i \neq j$ and $w_{i,-j}(t) = x_{i,-j}(t)x_{-i,j}(t^{-1})x_{i,-j}(t)$ for $i<j$. Hence we generate all the elementary matrices (**Table A5**) using only two generators x and w and an extra element w_l. Hence we generate a new subgroup $W\Omega(2l+1,p)$ of the orthogonal group $\mathrm{O}(2l+1,p)$, containing Ω, which is indeed a normal subgroup of $\mathrm{O}(2l+1,p)$. In our algorithm the output matrix is $d(\lambda) = \mathrm{diag}\left(1,1,\ldots,\lambda,1,\ldots,\lambda^{-1}\right)$. If $\lambda \in F_p^{\times 2}$, say $\lambda \equiv t^2(\mathrm{mod} p)$, here $t \equiv \lambda^{\frac{p+1}{4}}(\mathrm{mod} p)$, since $p \equiv 3$ (mod4). Then

$$\begin{aligned} d(\lambda) &= \mathrm{diag}\left(1,1,\ldots,t^2,1,\ldots,,t^{-2}\right) \\ &= w_{l-1,l}(1)\mathrm{diag}\left(1,1,\ldots,t^2,1,1,\ldots,,t^{-2},1\right)w_{l-1,l}(-1) \\ &= w_{l-1,l}(1)w_{l-1,l}(t)w_{l-1,l}(-1)w_{l-1,-l}(t)w_{l-1,-l}(-1)w_{l-1,l}(-1). \end{aligned}$$

Hence we generate $W\Omega(2l+1,p)$ using x, w and w_l.

Remark 5.1 *Let* $d(\zeta) = \mathrm{diag}\left(1,1,\ldots,\zeta,1,\ldots,\zeta^{-1}\right)$, *where* ζ *is non-square in* $\mathbb{F}_p^{\times}$. *The group* $\langle W\Omega, d(\zeta)\rangle$ *is the orthogonal group.*

5.4 Twisted orthogonal group $\mathbf{O}^{-}(2\boldsymbol{l},\boldsymbol{p})$

We use the following generators which we refer as Steinberg generators.

$$x = x_{1,2}(1),$$

$$x' = x_{-1,2}(1),$$

$$w = \begin{pmatrix} -I_2 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -I_{2l-3} & 0 \end{pmatrix},$$

$$w_l = I - e_{l,l} - e_{-l,-l} - e_{l,-l} - e_{-l,l},$$

$$x_1(t,s),\ \text{where } t \in \mathbb{F}_p^{\times}, s \in \mathbb{F}_p \text{ and } x_2.$$

In the context of MOR cryptosystem, we need to know how to go back and forth between these generators and elementary matrices (**Table A7**). The procedure is almost similar to the case of $\mathrm{O}^{+}(2l,p)$. Again, note that $x = x_{1,2}$, $x' = x_{-1,2}$, $x_1(t,s)$, and x_2 are elementary matrices. Thus, we just need to write w as a product of elementary matrices. However, computing w is fairly easy, just put this generator through our Gaussian elimination algorithm in Appendix A. Here we demonstrate the other way round, that is, how to write elementary matrices as a product of w, x, and x'. First, we compute $x_{1,i}(t) = w^{i-1}x_{1,2}(1)w^{-(i-1)}$ which is of length $2i-1$ for $2 \leq i \leq l$. Now we compute $x_{i,1}(t)$ using the relation $x_{i,1}(t) = w^{l-1}x_{1,i}(-t)w^{-(l-1)}$ for

$2 \leq i \leq l$, where $w^{l-1} = (-1)^{l-1}\begin{pmatrix} I_2 & 0 & 0 \\ 0 & 0 & I_{l-1} \\ 0 & I_{l-1} & 0 \end{pmatrix}$ and length of this SLP is $2(l-1) + 2i - 1$. Thus, we get $x_{i,1}(t)$ and $x_{1,i}(t)$, for $i = 2, \ldots, l$. Similarly we compute $x_{i,-1}(t)$ and $x_{-1,i}(t)$ using the relations $x_{-1,i}(t) = w^{i-1}x_{-1,2}(1)w^{-(i-1)}$ and $x_{i,-1}(t) = w^{l-1}x_{-1,i}(-t)w^{-(l-1)}$ for $2 \leq i \leq l$, and length of this SLP are $2i - 1$ and $2(l-1) + 2i - 1$, respectively. Next, we compute $x_{2,3}(t)$ using the commutator formula $x_{2,3}(t) = \left[x_{2,1}\left(\frac{t}{2}\right), x_{1,3}(1)\right]$, and length of this SLP is $4(l-1) + 8$. In what follows, we denote the length of SLPs by $L(\delta, i)$, where $\delta = j - i$ and $2 \leq i < j \leq l$.

$$\begin{array}{rcl} \delta = 1, & x_{i,j}(t) = & w^{i-1}x_{2,3}(t)w^{-(i-1)}, \\ \delta = 2, & x_{i,j}(t) = & \left[x_{i,j-1}(t), x_{j-1,j}(1)\right], \\ \delta = 3, & x_{i,j}(t) = & \left[x_{i,j-1}(t), x_{j-1,j}(1)\right], \\ \vdots & \vdots & \vdots \\ \delta = l-1, & x_{i,j}(t) = & \left[x_{i,j-1}(t), x_{j-1,j}(1)\right]. \end{array}$$

Here

$$L(\delta, i) = \begin{cases} 2i + 4(l-1) + 6 & \text{for } \delta = 1, \\ 2L(\delta - 1, i) + 4(i + \delta + 2(l-1) + 2) & \text{for } \delta = 2, 3, \ldots, l-2. \end{cases}$$

As $x_{j,i}(t) = w^{l-1}x_{i,j}(-t)w^{-(l-1)}$, so length of this SLP is $L(\delta, i) + 2(l-1)$. Hence, we get all $x_{i,j}(t)$ for $2 \leq i \neq j \leq l$ and the number of SLPs is $l + 2$. Next, we compute the remaining elementary matrices using the commutator formula and are listed in the table; let $r = l - 1$.

Elements	Indices	Equation (SLP)	Length	
$x_{1,-l}(t)$		$wx_{l-1,l}(t)w^{-1}$	$6(l-1) + 6$	
$x_{1,-i}(t)$	$2 \leq i \leq l-1$	$[x_{i,l}(t), x_{1,-l}(1)]$	$24(l-1) + 20$	$i = l-1$
			$2L(r-i, i) + 12(r+1)$	$i \neq l-1$
$x_{i,-j}(t)$	$2 \leq i \leq r-1$	$\left[x_{i,1}(t), x_{1,-j}(1)\right]$	$2L(i-1,1) + 4(7r+6) + 4L(r-j-\delta, j-\delta)$	$j < l-1$
	$(i+1 \leq j \leq l)$		$2L(i-1,1) + 4(7r+5)$	$j = l-1$
			$2L(i-1,1) + 10r + 6$	$j = l$

Thus, we have generated all $x_{i,-j}(t)$ for $i < j$. Now, using the formula $w^l x_{i,-j}(t)w^{-l} = x_{-i,j}(t)$, we get $x_{-i,j}(t)$ and the total number of SLPs required is $l + 6$. Now we know $w_{l-1} = w_l w_{l,l-1}(1)w_{l-1,-l}(1)$, so we generate w_{l-1}. Hence by induction we can generate $w_i = w_{i+1}w_{i+1,i}(1)w_{i,-(i+1)}(1)$, for $i = l-1, \ldots, 2$. Here $w_{i,j}(t) = x_{i,j}(t)x_{j,i}(-t^{-1})x_{i,j}(t)$, for $i \neq j$, and $w_{i,-j}(t) = x_{i,-j}(t)x_{-i,j}(t^{-1})x_{i,-j}(t)$, for $i < j$. Hence we generate all the elementary matrices defined in **Table A7** using generators x, x', $x_1(t,s)$, x_2, and w and an extra element w_l. In our algorithm the output matrix is $d(\lambda) = \text{diag}\left(1, 1, 1, \ldots, \lambda, 1, \ldots, \lambda^{-1}\right)$. If $\lambda \in F_p^{\times 2}$, say $\lambda \equiv t^2 (\text{mod}\, p)$, here $t \equiv \lambda^{\frac{p+1}{4}} (\text{mod}\, p)$, since $p \equiv 3 \ (\text{mod}\ 4)$.

$$\begin{aligned} \text{Then } d(\lambda) &= \text{diag}\left(1, 1, 1, \ldots, t^2, 1, \ldots, , t^{-2}\right) \\ &= w_{l-1,l}(1)\text{diag}\left(1, 1, 1, \ldots, t^2, 1, 1, \ldots, , t^{-2}, 1\right)w_{l-1,l}(-1) \\ &= w_{l-1,l}(1)w_{l-1,l}(t)w_{l-1,l}(-1)w_{l-1,-l}(t)w_{l-1,-l}(-1)w_{l-1,l}(-1). \end{aligned}$$

Remark 5.2 *Let $d(\zeta) = \mathrm{d}iag\left(1, 1, 1, ..., \zeta, 1, ..., \zeta^{-1}\right)$, where ζ is non-square in $\mathbb{F}_p^\times$. Then as a consequence of our Gaussian elimination algorithm in Appendix A, we can see that x, x', $x_1(t, s)$, x_2,w and w_l along with $d(\zeta)$ generate the twisted orthogonal group.*

6. Conclusion

This section is similar to ([6], Section 8). A useful public-key cryptosystem is a delicate dance between speed and the security. So one must talk about speed along with security.

The implementation of the MOR cryptosystem that we have in mind uses the row-column operations. Let $\langle g_1, g_2, ..., g_s \rangle$ be a set of generators for the orthogonal or symplectic group as described before. As is the custom with a MOR cryptosystem, the automorphisms ϕ and ϕ^m are presented as action on generators, i.e., we have $\phi\left(g_i\right)$ and $\phi^m\left(g_i\right)$ as matrices for $i = 1, 2, ..., s$.

To encrypt a message in this MOR cryptosystem, we compute ϕ^r. We do that by *square-and-multiply* algorithm. For this implementation, squaring and multiplying is almost the same. So we will refer to both squaring and multiplication as multiplication. Note that multiplication is composed of automorphisms.

The implementation that we describe in this chapter can work in parallel. Each instance computes $\phi^r\left(g_i\right)$ for $i = 1, 2, ..., s$. First thing that we do is write the matrix of $\phi\left(g_i\right)$ as a word in generators. So essentially the map ϕ becomes a map $g_i \mapsto w_i$ where w_i is a word in generators of some fixed length. Then multiplication becomes essentially a replacement, replace all instances of g_i by w_i. This can be done very fast. However, the length of the replaced word can become very large. The obvious question is how soon are we going to write this word as a matrix. This is a difficult question to answer at this stage and depends on available computational resources.

Once we decide how often we change back to matrices, how are we going to change back to matrices? There can be a fairly easy *time-memory* trade-offs. Write all words up to a fixed length and the corresponding matrix as a pre-computed table and use this table to compute the matrices. Once we have matrices, we can multiply them together to generate the final output. There are also many obvious relations among the generators of these groups. One can just store and use them. The best strategy for an efficient implementation is yet to be determined. It is clear now that there are many interesting and novel choices.

The benefits of this MOR cryptosystem are:

This can be implemented in parallel easily.

This implementation does not depend on the size of the characteristic of the field. This is an important property in light of Joux's recent improvement of the index-calculus attacks [11].

For parameters and complexity analysis of this cryptosystem, we refer to ([6], Section 8). Assume that we take a prime of size 2^{160} and we are using two generators presentation of ϕ for the even-orthogonal group. Then the security is the discrete logarithm problem in $\mathbb{F}_{p^{d^2}}$. Now if we take $d = 4$, then the security is better than $\mathbb{F}_{2^{2560}}$. Our key-size is about 8000 bits. Comparing with Monico ([10], Section 7), where he says an ElGamal will have about 6080 bits, our system is quite comparable. Moreover, the MOR cryptosystem is better suited to handle large primes and can be easily parallelized.

Acknowledgements

We are thankful to the editor and referees for their valuable comments which has improved the paper substantially. This work was supported by a SERB research grant. This chapter contains part of the PhD thesis of the first and the third author, directed by the second and the fourth author at IISER Pune.

Appendix A. Solving the word problem in G

In computational group theory, one is always looking for algorithms that solve the word problem. When G is a special linear group, one has a well-known algorithm to solve the word problem—the Gaussian elimination algorithm. One observes that the effect of multiplying an element of the special linear group by an elementary matrix (also known as elementary transvection) from left or right is either a row or a column operation, respectively. Using this algorithm one can start with any matrix $g \in \mathrm{SL}(l+1, k)$ and get to the identity matrix, thus writing g as a product of elementary matrices ([18], Proposition 6.2). One of the **objective of this appendix** is to discuss a similar algorithm for orthogonal and symplectic groups, with a set of generators that we will call **elementary matrices in their respective groups**. Similar algorithms can be found in the works of Brooksbank [19, 20] and Costi [21]. However, we have no restrictions on the cardinality or characteristic of the field k.

We first describe the elementary matrices and the row-column operations for the respective groups. These row-column operations are nothing but multiplication by elementary matrices from left and right, respectively. Here elementary matrices used are nothing but Chevalley generators which follows from the theory of Chevalley groups.

The basic idea of the algorithm is to use the fact that multiplying any orthogonal matrix by any one of the generators enables us to perform row or column operations. The relation ${}^{T}g\beta g = \beta$ gives us some compact relations among the blocks of g which can be used to make the algorithm faster. To make the algorithm simple, we will write the algorithm for $\mathrm{O}(2l+1, k)$, $\mathrm{O}^{+}(2l, k)$, and $\mathrm{O}^{-}(2l, k)$ separately.

A.1 Groups in which Gaussian elimination works

- Symplectic groups: Since all non-degenerate skew-symmetric bilinear forms are equivalent ([22], Corollary 2.12), we have a Gaussian elimination algorithm for all symplectic groups over an arbitrary field.

- Orthogonal groups:

    - Since non-degenerate symmetric bilinear forms over a finite field of odd characteristics are classified ([22], p. 79) according to the β (see Section 3), we have a Gaussian elimination algorithm for all orthogonal groups over a finite field of odd characteristics.

    - Since non-degenerate quadratic forms over a perfect field of even characteristics can be classified ([23], p. 10) according to quadratic forms $Q(x)$ defined in ([24], Section 4.2), we have a Gaussian elimination algorithm for all orthogonal groups over a perfect field of even characteristics.

- Furthermore, we have Gaussian elimination algorithm for orthogonal groups that are given by the above bilinear forms or quadratic forms over arbitrary fields. This algorithm also works for bilinear or quadratic forms that are equivalent to the above forms.

A.2 Gaussian elimination for matrices of even size—orthogonal group $\mathrm{O}^{+}(\boldsymbol{d}, \boldsymbol{k})$ and symplectic group

Recall that the bilinear forms β are the following:

- For symplectic group, $\mathrm{Sp}(d, k)$, $d = 2l$, and $\beta = \begin{pmatrix} 0 & I_l \\ -I_l & 0 \end{pmatrix}$.

- For orthogonal group, $\mathrm{O}^{+}(d, k)$, $d = 2l$, and $\beta = \begin{pmatrix} 0 & I_l \\ I_l & 0 \end{pmatrix}$.

Note that any isometry g satisfies ${}^T g \beta g = \beta$. The main reason our algorithm works is the following: Recall that a matrix $g = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$, where A, B, C, and D are matrices of size l, is orthogonal or symplectic if ${}^T g \beta g = \beta$ for the respective β. After some usual calculations, for orthogonal group it becomes

$$\begin{pmatrix} {}^T CA + {}^T AC & {}^T CB + {}^T AD \\ {}^T DA + {}^T BC & {}^T DB + {}^T BD \end{pmatrix} = \begin{pmatrix} 0 & I_l \\ I_l & 0 \end{pmatrix} \tag{A.1}$$

The above equation implies among other things, ${}^T CA + {}^T AC = 0$. This implies that ${}^T AC$ is skew-symmetric. In an almost identical way, one can show, if g is symplectic, ${}^T AC$ is symmetric. The working principle of our algorithm is simple—use the symmetry of ${}^T AC$. The problem is, for arbitrary A and C, it is not easy to use this symmetry. In our case we were able to reduce A to a diagonal matrix, and then it is relatively straightforward to use this symmetry. We will explain the algorithm in details later. First of all, let us describe the elementary matrices and the row-column operations for orthogonal and symplectic groups. The genesis of these elementary matrices lies in the Chevalley basis of simple Lie algebras. We will not go into details of Chevalley's theory in this appendix. Furthermore, we do not need to, the algorithm that we produce will show that these elementary matrices are generators for the respective groups.

Next we present the elementary matrices for the respective groups and then the row-column operations in a tabular form.

A.2.1 Elementary matrices (Chevalley generators) for orthogonal group $\mathrm{O}^{+}(\boldsymbol{d}, \boldsymbol{k})$ of even size

Following the theory of root system in a simple Lie algebra, we index rows by $1, 2, \ldots, l, -1, -2, \ldots, -l$. For $t \in k$, the elementary matrices are defined as follows (**Tables A1** and **A2**):

Let us note the effect of multiplying g by elementary matrices. We write

Char(k)		**Elementary matrices**	
	$x_{i,j}(t)$	$I + t\left(e_{i,j} - e_{-j,-i}\right)$	$i \neq j$
Both	$x_{i,-j}(t)$	$I + t\left(e_{i,-j} - e_{j,-i}\right)$	$i < j$
	$x_{-i,j}(t)$	$I + t\left(e_{-i,j} - e_{-j,i}\right)$	$i < j$
	w_i	$I - e_{i,i} - e_{-i,-i} + e_{i,-i} + e_{-i,i}$	$1 \leq i \leq l$

Table A1.
Elementary matrices for $O^+(2l,k)$.

	Row operations		**Column operations**
ER1	ith↦ith $+ tj$th row	EC1	jth↦jth $+ ti$th column
	$-j$th↦ $-j$th $- t(-i)$th row		$-i$th↦ $-i$th $- t(-j)$th column
ER2	ith↦ith $+ t(-j)$th row	EC2	$-i$th↦ $-i$th $- tj$th column
	jth↦jth $- t(-i)$th row		$-j$th↦ $-j$th $+ ti$th column
ER3	$-i$th↦ $-i$th $- tj$th row	EC3	jth↦jth $+ t(-i)$th column
	$-j$th↦ $-j$th $+ ti$th row		ith↦ith $- t(-j)$th column
w_i	Interchange ith and $(-i)$th row		Interchange ith and $(-i)$th column

Table A2.
The row-column operations for $O^+(2l,k)$.

Char(k)		**Elementary matrices**	
	$x_{i,j}(t)$	$I + t\left(e_{i,j} - e_{-j,-i}\right)$	$i \neq j$
Both	$x_{i,-j}(t)$	$I + t\left(e_{i,-j} + e_{j,-i}\right)$	$i < j$
	$x_{-i,j}(t)$	$I + t\left(e_{-i,j} + e_{-j,i}\right)$	$i < j$
	$x_{i,-i}(t)$	$I + te_{i,-i}$	$1 \leq i \leq l$
	$x_{-i,i}(t)$	$I + te_{-i,i}$	$1 \leq i \leq l$

Table A3.
Elementary matrices for $Sp(2l,k)$.

$g \in \mathrm{O}^+(2l,k)$ as $g = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$, where A, B, C, and D are $l \times l$ matrices.

A.2.2 Elementary matrices (Chevalley generators) for symplectic group

For $t \in k$, the elementary matrices are defined as follows (**Table A3**):

Let us note the effect of multiplying g by elementary matrices. We write $g \in \mathrm{Sp}(2l,k)$ as $g = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$, where A, B, C, and D are $l \times l$ matrices (**Table A4**).

A.2.3 Gaussian elimination for $\mathbf{Sp}(2l,k)$ and $\mathbf{O}^+(2l,k)$

Step 1: Use ER1 and EC1 to make A into a diagonal matrix. This makes A into a diagonal matrix and changes other matrices A, B, C, and D. For the sake of notational convenience, we keep calling these changed matrices as A, B, C, and D as well.

	Row operations		Column operations
ER1	ith↦ith $+ tj$th row	EC1	jth↦jth $+ ti$th column
	$-j$th↦ $-j$th $+ t(-i)$th row		$-i$th↦ $-i$th $+ t(-j)$th column
ER2	ith↦ith $+ t(-j)$th row	EC2	$-i$th↦ $-i$th $+ tj$th column
	jth↦jth $+ t(-i)$th row		$-j$th↦ $-j$th $+ ti$th column
ER3	$-i$th↦ $-i$th $+ tj$th row	EC3	jth↦jth $+ t(-i)$th column
	$-j$th↦ $-j$th $+ ti$th row		ith↦ith $+ t(-j)$th column
ER1a	ith↦ith $+ t(-i)$th row	EC1a	$-i$th↦ $-i$th $+ ti$th column
ER2a	$-i$th↦ $-i$th $+ ti$th row	EC2a	ith↦ith $+ t(-i)$th column
w_i	Interchange ith and $(-i)$th rows		Interchange ith and $(-i)$th columns
	with a sign change in the ith row		with a sign change in the ith column

Table A4.
The row-column operations for symplectic groups.

Step 2: There are two possibilities. One, the diagonal matrix A is of full rank, and two, the diagonal matrix A is of rank $\mathfrak{r}$ less than l. This is clearly identifiable by looking for zeros in the diagonal of A.

Step 3: Make $\mathfrak{r}$ rows of C, corresponding to the non-zero entries in the diagonal of A zero by using ER3. If $\mathfrak{r} = l$, we have C as zero matrix. If not let us assume that ith row is zero in A. Then we interchange the i^{th} row with the $-i^{\text{th}}$ row in g. We do this for all zero rows in A. The new C is a zero matrix. We claim that the new A must have a full rank. This follows from Equation A.1; in particular ${}^{T}CB + {}^{T}AD = I_l$. If C is zero matrix, then A is invertible. Now make A a diagonal matrix by using Step 1. Then one can make A a matrix of the form $\text{diag}(1, \ldots, 1, \lambda)$, where $\lambda \in k^{\times}$ using ER1 ([18], Proposition 6.2). Once A is diagonal and C a zero matrix, the equation ${}^{T}CB + {}^{T}AD = I_l$ makes D a diagonal matrix of full rank.

Step 4: Use ER2 to make B a zero matrix. The matrix g becomes a diagonal matrix of the form

$\text{diag}\left(1, \ldots, 1, \lambda, 1, \ldots, 1, \lambda^{-1}\right)$, where $\lambda \in k^{\times}$.

Step 5: (Only for symplectic groups) Reduce the λ to 1 using Lemma A.1.

Lemma A.1 *For* $\text{Sp}(2l, k)$, *the element* $\text{diag}\left(1, \ldots, 1, \lambda, 1, \ldots, 1, \lambda^{-1}\right)$ *is a product of elementary matrices.*

Proof. Observe that $(I + \lambda e_{l,-l})\left(I - \lambda^{-1} e_{-l,l}\right)(I + \lambda e_{l,-l}) = I - e_{l,l} - e_{-l,-l} + \lambda e_{l,-l} - \lambda^{-1} e_{-l,l}$ and denote it by $w_l(\lambda)$, and then the diagonal element is $w_l(\lambda) w_l(-1)$.

Remark A.1 *As we saw in the above algorithm, we will have to interchange* i^{th} *and* $-i^{th}$ *rows for* $i = 1, 2, \ldots, l$. *This can be done by pre-multiplying with a suitable matrix.*

Let I be the $2l \times 2l$ identity matrix over k. To swap ith and $-i$th row in $\text{O}^{+}(2l, k)$, swap ith and $-i$th rows in the matrix I. We will call this matrix w_i. It is easy to see that this matrix w_i is in $\text{O}^{+}(2l, k)$ and is of determinant -1. Pre-multiplying with w_i does the row interchange we are looking for.

In the case of symplectic group $\text{Sp}(2l, k)$, we again swap two rows ith and $-i$th in I. However we do a sign change in the ith row and call it w_i. Simple computation with our chosen β shows that the above matrices are in $\text{O}^{+}(2l, k)$ and $\text{Sp}(2l, k)$, respectively.

However there is one difference between orthogonal and symplectic groups. In symplectic group, w_i can be generated by elementary matrices because $w_i = x_{i,-i}(1)x_{-i,i}(-1)x_{i,-i}(1)$. In the case of orthogonal groups, that is not the case. This is clear that the elementary matrices come from the Chevalley generators and those generates Ω, the commutator of the orthogonal group. All matrices in Ω have determinant 1. However w_i has determinant -1. So we must add w_i as an elementary matrix for $\mathrm{O}^+(2l, k)$.

Remark A.2 *This algorithm proves every element in the symplectic group is of determinant 1. Note the elementary matrices for the symplectic group are of determinant 1, and we have an algorithm to write any element as product of elementary matrices. So this proves that the determinant is 1.*

Remark A.3 *This algorithm proves if X is an element of a symplectic group then so is* ${}^{T}X$. *The argument is similar to the above; here we note that the transpose of an elementary matrix in symplectic groups is an elementary matrix.*

A.3 Gaussian elimination for matrices of odd size—the odd-orthogonal group

In this case, matrices are of odd size and there is only one family of group to consider; it is the odd-orthogonal group $\mathrm{O}(2l+1, k)$. This group will be referred to as the odd-orthogonal group.

A.3.1 Elementary matrices (Chevalley generators) for $O(2l+1, k)$

Following the theory of Lie algebra, we index rows by $0, 1, \ldots, l, -1, \ldots, -l$. These elementary matrices are listed in **Table A5**.

Elementary matrices for the odd-orthogonal group in even characteristics differ from that of odd characteristics. In above table we made that distinction and listed them separately in different rows according to the characteristics of k. If $\mathrm{char}(k)$ is even, we can construct the elements w_i, which interchanges the ith row with $-i^{th}$ row as follows:

$$w_i = (I + e_{0,i} + e_{-i,i})(I + e_{0,-i} + e_{i,-i})(I + e_{0,i} + e_{-i,i}) = I + e_{i,i} + e_{-i,-i} + e_{i,-i} + e_{-i,i}.$$

Otherwise, we can construct w_i, which interchanges the ith row with $-i^{th}$ row with a sign change in i^{th}, $-i^{th}$ and 0^{th} row in odd-orthogonal group as follows:

Char(k)		**Elementary matrices**	
Both	$x_{i,j}(t)$	$I + t(e_{i,j} - e_{-j,-i})$	$i \neq j$
	$x_{i,-j}(t)$	$I + t(e_{i,-j} - e_{j,-i})$	$i < j$
	$x_{-i,j}(t)$	$I + t(e_{-i,j} - e_{-j,i})$	$i < j$
Odd	$x_{i,0}(t)$	$I + t(2e_{i,0} - e_{0,-i}) - t^2 e_{i,-i}$	$1 \leq i \leq l$
	$x_{0,i}(t)$	$I + t(-2e_{-i,0} + e_{0,i}) - t^2 e_{-i,i}$	$1 \leq i \leq l$
Even	$x_{i,0}(t)$	$I + te_{0,-i} + t^2 e_{i,-i}$	$1 \leq i \leq l$
	$x_{0,i}(t)$	$I + te_{0,i} + t^2 e_{-i,i}$	$1 \leq i \leq l$

Table A5.
Elementary matrices for $O(2l+1, k)$.

$$w_i = x_{0,i}(-1)x_{i,0}(1)x_{0,i}(-1) = I - 2e_{0,0} - e_{i,i} - e_{-i,-i} - e_{i,-i} - e_{-i,i}.$$

The Gaussian elimination algorithm for $\mathrm{O}(2l+1,k)$ follows the earlier algorithm for symplectic and even-orthogonal group closely, except that we need to take care of the zero row and the zero column. We write an element $g \in O(2l+1,k)$ as $g = \begin{pmatrix} \alpha & X & Y \\ E & A & B \\ F & C & D \end{pmatrix}$, where A, B, C, and D are $l \times l$ matrices, X and Y are $1 \times l$ matrices, E and F are $l \times 1$ matrices, $\alpha \in k$ and $\beta = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 0 & I_l \\ 0 & I_l & 0 \end{pmatrix}$. Then from the condition ${}^T g\beta g = \beta$, we get the following relations:

$$2{}^T XX + {}^T AC + {}^T CA = 0 \tag{A.2}$$

$$2\alpha{}^T X + {}^T AF + {}^T CE = 0 \tag{A.3}$$

$$2\alpha Y + {}^T ED + {}^T FB = 0 \tag{A.4}$$

$$2{}^T XY + {}^T AD + {}^T CB = I_l \tag{A.5}$$

Let us note the effect of multiplying g by elementary matrices (**Table A6**).

	Row operations		**Column operations**
ER1	ith$\mapsto i$th $+\, tj$th row	EC1	jth$\mapsto j$th $+\, ti$th column
(both)	$-j$th$\mapsto -j$th $-\, t(-i)$th row	(both)	$-i$th$\mapsto -i$th $-\, t(-j)$th column
ER2	ith$\mapsto i$th $+\, t(-j)$th row	EC2	$-i$th$\mapsto -i$th $-\, tj$th column
(both)	jth$\mapsto j$th $-\, t(-i)$th row	(both)	$-j$th$\mapsto -j$th $+\, ti$th column
ER3	$-i$th$\mapsto -i$th $-\, tj$th row	EC3	jth$\mapsto j$th $+\, t(-i)$th column
(both)	$-j$th$\mapsto -j$th $+\, ti$th row	(both)	ith$\mapsto i$th $-\, t(-j)$th column
ER4	0th$\mapsto 0$th $-\, t(-i)$th row	EC4	0th$\mapsto 0$th $+\, 2ti$th column
(odd)	ith$\mapsto i$th $+\, 2t0$th $-\, t^2(-i)$th row	(odd)	$(-i)$th$\mapsto(-i)$th $-\, t0$th $-\, t^2i$th column
ER5	0th$\mapsto 0$th $+\, ti$th row	EC5	0th$\mapsto 0$th $-\, 2t(-i)$th column
(odd)	$(-i)$th$\mapsto(-i)$th $-\, 2t0$th $-\, t^2i$th row	(odd)	ith$\mapsto i$th $+\, t0$th $-\, t^2(-i)$th column
ER6	0th$\mapsto 0$th $+\, t(-i)$th row	EC6	$(-i)$th$\mapsto(-i)$th $+\, t0$th $+\, t^2i$th column
(even)	ith$\mapsto i$th $+\, t^2(-i)$th row	(even)	
ER7	0th$\mapsto 0$th $+\, ti$th row	EC7	ith$\mapsto i$th $+\, t0$th $+\, t^2(-i)$th column
(even)	$(-i)$th$\mapsto(-i)$th $+\, t^2i$th row	(even)	
w_i	Interchange ith and $(-i)$th rows	w_i	Interchange ith and $(-i)$th column
(odd)	with a sign change in ith, $-i$th and 0th rows	(odd)	with a sign change in ith, $-i$th and 0th columns
w_i (even)	Interchange ith and $(-i)$th row	w_i (even)	Interchange ith and $(-i)$th column

Table A6.
The row-column operations for $O(2l+1,k)$.

A.3.2 Gaussian elimination for $\mathbf{O}(2\boldsymbol{l}+\mathbf{1},\boldsymbol{k})$

Step 1: Use ER1 and EC1 to make A into a diagonal matrix, but in the process, it changes other matrices A, B, C, D, E, F, X, and Y. For the sake of notational convenience, we keep calling these changed matrices as A, B, C, D, E, F, X, and Y as well.

Step 2: Now there will be two cases depending on the rank $\mathfrak{r}$ of matrix A. The rank of A can be easily determined using the number of non-zero diagonal entries. Use ER3 and non-zero diagonal entries of A to make corresponding $\mathfrak{r}$ rows of C zero.

1. If $\mathfrak{r}=l$ then C becomes zero matrix.
2. If $\mathfrak{r}<l$ then *interchange* all zero rows of A with corresponding rows of C using w_i so that the new C becomes a zero matrix.

Once C becomes zero, note that Relation A.2 if $\text{char}(k)$ is odd or Relation $Q(g(v))=Q(v)$ if $\text{char}(k)$ is even guarantees that X becomes zero. Relation A.5 guarantees that A has full rank l which also makes D a diagonal with full rank l. Thus Relation A.3 shows that F becomes zero as well. Then use Step 1 to reduce $A=\text{diag}(1,\ldots,1,\lambda)$, where $\lambda\in k^{\times}$.

Step 3: Now if $\text{char}(k)$ is even, then Relation A.4 guarantees that E becomes zero as well. If $\text{char}(k)$ is odd, then use ER4 to make E a zero matrix.

Step 4: Use ER2 to make B a zero matrix. For $\text{char}(k)$ even the relation $Q(g(v))=Q(v)$ guarantees that Y is a zero matrix, and for $\text{char}(k)$ odd Relation A.4 implies that Y becomes zero.

Thus the matrix g reduces to $\text{diag}(\pm 1, 1,\ldots,\lambda,1,\ldots,\lambda)$, where $\lambda\in k^{\times}$.

Remark A.4 *Let k be a perfect filed of characteristics 2. Note that we can write the diagonal matrix* $\text{diag}\left(1,\ldots,1,\lambda,1,\ldots,1,\lambda^{-1}\right)$ *as a product of elementary matrices as follows:*

$$\text{diag}\left(1,\ldots,1,\lambda,1,\ldots,1,\lambda^{-1}\right)=x_{l,-l}(t)x_{-l,l}(-t^{-1})x_{l,-l}(t), \quad \text{where} \quad t^2=\lambda,$$

and hence we can reduce the matrix g to identity.

A.4 Gaussian elimination in twisted orthogonal groups

In this section we present a Gaussian elimination algorithm for twisted orthogonal groups. The size of the matrix is even; the bilinear form used is c′ from Section 3.

A.4.1 Elementary matrices (Chevalley generators) for twisted orthogonal groups $\mathbf{O}^{-}(2\boldsymbol{l},\boldsymbol{k})$

In this section, we describe row-column operations for twisted Chevalley groups. These groups are also known as the Steinberg groups. An element $g\in \text{O}^{-}(2l,k)$ is denoted as $g=\begin{pmatrix} A_0 & X & Y \\ E & A & B \\ F & C & D \end{pmatrix}$, where A, B, C, and D are $(l-1)\times(l-1)$ matrices, X and Y are $2\times(l-1)$ matrices, E and F are $(l-1)\times 2$ matrices, and A_0 is a 2×2 matrix. In the Gaussian elimination algorithm that we discuss, we reduce X, Y, E, F, B, and C to zero and A and D to diagonal matrices.

However, unlike the previous cases, we were unable to reduce A_0 to an identity matrix. However, for odd characteristics we were able to reduce A_0 to a two-parameter subgroup.

We now talk about the output of the algorithm. In the output we will have a 2×2 block (also called A_0) which will satisfy ${}^T A_0 \beta_0 A_0 = \beta_0$, where $\beta_0 = \begin{pmatrix} 1 & 0 \\ 0 & \epsilon \end{pmatrix}$ for odd characteristics and ε is a non-square. Then A_0 is a orthogonal matrix given by the bilinear form β_0. Now if we write $A_0 = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, then we get the following equations:

$$a^2 + c^2\epsilon = 1, \quad ab + cd\epsilon = 0, \quad b^2 + d^2\epsilon = \epsilon.$$

Considering the fact that $\det(A_0) = \pm 1$, one more equation $ad - bc = \pm 1$ and this leads to two cases either $a = d$ and $b = -c\epsilon$ or $a = -d$ and $b = c\epsilon$. Recall that, since ϵ is not a square, $d \neq 0$. Then if $c = 0$, then there are four choices for A_0 and these are $A_0 = \begin{pmatrix} \pm 1 & 0 \\ 0 & \pm 1 \end{pmatrix}$.

To summarize, the output of the algorithm A_0 will have one of the following forms

$$\begin{pmatrix} t & -s\epsilon \\ s & t \end{pmatrix} \text{or} \quad \begin{pmatrix} t & s\epsilon \\ s & -t \end{pmatrix}, \text{where} \quad t^2 + s^2\epsilon = 1, \tag{A.6}$$

and $t \in k^\times$, $s \in k$, and ϵ are non-square. There are now two ways to describe the algorithm: one is to leave A_0 as it is in the output of the algorithm, and the other is to include these matrices as generators. For the purpose of uniform exposition, we chose the latter and included the following two generators

Char(k)		**Elementary matrices**	
	$x_{i,j}(t)$	$I + t(e_{i,j} - e_{-j,-i})$	$i \neq j$
Both	$x_{i,-j}(t)$	$I + t(e_{i,-j} - e_{j,-i})$	$i < j$
	$x_{-i,j}(t)$	$I + t(e_{-i,j} - e_{-j,i})$	$i < j$
	w_i	$I - e_{i,i} - e_{-i,-i} + e_{i,-i} + e_{-i,i}$	$2 \leq i \leq l$
	$x_{i,1}(t)$	$I + t(e_{1,i} - 2e_{-i,1}) - t^2 e_{-i,i}$	$2 \leq i \leq l$
	$x_{1,i}(t)$	$I + t(-e_{1,-i} + 2e_{i,1}) - t^2 e_{i,-i}$	$2 \leq i \leq l$
	$x_{i,-1}(t)$	$I + t(e_{-1,i} - 2\varepsilon e_{-i,-1}) - \varepsilon t^2 e_{-i,i}$	$2 \leq i \leq l$
Odd	$x_{-1,i}(t)$	$I + t(-e_{-1,-i} + 2\varepsilon e_{i,-1}) - \varepsilon t^2 e_{i,-i}$	$2 \leq i \leq l$
	$x_1(t,s)$	$I + (t-1)e_{1,1} - (t+1)e_{-1,-1} + s(e_{-1,1} + \varepsilon e_{1,-1})$	$t^2 + \varepsilon s^2 = 1$
	x_2	$I - 2e_{-1,-1}$	
	$x_{1,-i}(t)$	$I + te_{1,-i} + te_{i,-1} + \alpha t^2 e_{i,-i}$	$2 \leq i \leq l$
Even	$x_{-1,-i}(t)$	$I + te_{-1,-i} + te_{i,1} + \alpha t^2 e_{i,-i}$	$2 \leq i \leq l$
	x_{A_0}	$I + (t-1)e_{1,1} + (s-1)e_{-1,-1} + pe_{1,-1} + re_{-1,1}$	$ts + pr = 1$

Table A7.
Elementary matrices for $O^-(2l,k)$.

$$x_1(t,s) = I + (t-1)e_{1,1} - (t+1)e_{-1,-1} + s(e_{-1,1} + \epsilon e_{1,-1}); \quad t^2 + \epsilon s^2 = 1,$$
$$x_2 = I - 2e_{-1,-1},$$

in the list of elementary matrices in **Table A7**. In the case of even characteristics, no such reduction is possible, and we included the matrix $\begin{pmatrix} t & p \\ r & s \end{pmatrix}$ in the list of generators with the condition that the determinant is 1.

The elementary matrices for $\mathrm{O}^-(2l,k)$ depend on the characteristics of k. We describe them separately in the following table. Let α be an Arf-invariant, $2 \le i, j \le l$ and $t \in K$ and $\xi \in k^\times$.

Let us note the effect of multiplying g by elementary matrices. Elementary matrices for the twisted orthogonal group in even characteristics differ from that of odd characteristics, so in the following tables (**Tables A8** and **A9**), we made that distinction and listed them separately in different rows according to the characteristics of k.

	Row operations
ER1 (both)	ith$\mapsto i$th $+ tj$th row and $-j$th$\mapsto -j$th $- t(-i)$th row
ER2 (both)	ith$\mapsto i$th $+ t(-j)$th row and jth$\mapsto j$th $- t(-i)$th row
ER3 (both)	$-i$th$\mapsto -i$th $- tj$th row and $-j$th$\mapsto -j$th $+ ti$th row
ER4 (odd)	1st$\mapsto$1st $- t(-i)$th row and ith$\mapsto i$th $+ 2t$1st $- t^2(-i)$th row
ER5 (odd)	1st$\mapsto$1st $+ ti$th row and $(-i)$th$\mapsto(-i)$th $- 2t$1st $- t^2i$th row
ER6 (odd)	(-1)th$\mapsto(-1)$th $- t(-i)$th row and ith$\mapsto i$th $+ 2\varepsilon t(-1)$th $- \varepsilon t^2(-i)$th row
ER7 (odd)	(-1)th$\mapsto(-1)$th $+ ti$th row and $(-i)$th$\mapsto(-i)$th $- 2\varepsilon t(-1)$th $- \varepsilon t^2 i$th row
ER8 (even)	1st$\mapsto$1st $+ t(-i)$th row and ith$\mapsto i$th $+ t(-1)$th $+ \alpha t^2(-i)$th row
ER9 (even)	(-1)th$\mapsto(-1)$th $+ t(-i)$th row and ith$\mapsto i$th $+ t$1st $+ \alpha t^2(-i)$th row
w_i (both)	Interchange ith and $(-i)$th row

Table A8.
The row operations for $O^-(2l,k)$.

	Column operations
EC1(both)	jth$\mapsto j$th $+ ti$th column and $-i$th$\mapsto -i$th $- t(-j)$th column
EC2 (both)	$-i$th$\mapsto -i$th $- tj$th column and $-j$th$\mapsto -j$th $+ ti$th column
EC3 (both)	jth$\mapsto j$th $+ t(-i)$th column and ith$\mapsto i$th $- t(-j)$th column
EC4 (odd)	1st$\mapsto$1st $+ 2ti$th column and $(-i)$th$\mapsto(-i)$th $- t$1st $- t^2i$th column
EC5 (odd)	1st$\mapsto$1st $- 2t(-i)$th column and ith$\mapsto i$th $+ t$1st $- t^2(-i)$th column
EC6 (odd)	(-1)th$\mapsto(-1)$th $+ (2\varepsilon t)i$th column and $(-i)$th$\mapsto(-i)$th $- t(-1)$th $- \varepsilon t^2 i$th column
EC7 (odd)	(-1)th$\mapsto(-1)$th $- 2\varepsilon t(-i)$th column and ith$\mapsto i$th $+ t(-1)$th $- \varepsilon t^2(-i)$th column
EC8 (even)	(-1)th$\mapsto(-1)$th $+ ti$th column and $(-i)$th$\mapsto(-i)$th $+ t$1st $+ \alpha t^2 i$th column
EC9 (even)	1st$\mapsto$1st $+ ti$th column and $(-i)$th$\mapsto(-i)$th $+ t(-1)$th $+ \alpha t^2 i$th column
w_i (both)	Interchange ith and $(-i)$th column

Table A9.
The column operations for $O^-(2l,k)$.

Note that any isometry g satisfies ${}^{T}g\beta g = \beta$. The main reason the following algorithm works is the closed condition ${}^{T}g\beta g = \beta$ which gives the following relations:

$$ {}^{T}A_0\beta_0 A_0 + {}^{T}FE + {}^{T}EF = \beta_0, \tag{A.7}$$

$$ {}^{T}A_0\beta_0 X + {}^{T}FA + {}^{T}EC = 0, \tag{A.8}$$

$$ {}^{T}A_0\beta_0 Y + {}^{T}FB + {}^{T}ED = 0, \tag{A.9}$$

$$ {}^{T}X\beta_0 X + {}^{T}CA + {}^{T}AC = 0, \tag{A.10}$$

$$ {}^{T}X\beta_0 Y + {}^{T}CB + {}^{T}AD = I_{l-1}. \tag{A.11}$$

A.4.2 The Gaussian elimination algorithm for $\mathbf{O}^{-}(2l, k)$

Step 1: Use ER1 and EC1 to make A into a diagonal matrix, but in the process, it changes other matrices A_0, A, B, C, D, E, F, X, and Y. For the sake of notational convenience, we keep calling these changed matrices as A_0, A, B, C, D, E, F, X, and Y as well.

Step 2: Now there will be two cases depending on the rank $\mathfrak{r}$ of the matrix A. The rank of A can be easily determined by the number of non-zero diagonal entries.

Step 3: Use ER3 and non-zero diagonal entries of A to make corresponding $\mathfrak{r}$ rows of C zero.

- If $\mathfrak{r} = l - 1$ then C becomes zero matrix.
- If $\mathfrak{r} < l - 1$ then interchange all zero rows of A with corresponding rows of C using w_i, so that the new C becomes a zero matrix.
- Once C becomes zero one, can note that the relation ${}^{T}X\beta_0 X + {}^{T}CA + {}^{T}AC = 0$ if $\mathrm{char}(k)$ is odd or the relation $Q(g(v)) = Q(v)$ and the fact that $\alpha t^2 + t + \alpha$ is irreducible when $\mathrm{char}(k)$ is even guarantees that X becomes zero. Then the relation ${}^{T}X\beta_0 Y + {}^{T}CB + {}^{T}AD = I_{l-1}$ guarantees that A has full rank $l - 1$ which also makes D a diagonal with full rank, and the relation ${}^{T}A_0\beta_0 X + {}^{T}FA + {}^{T}EC = 0$ shows that F is zero. Now we diagonalize A again to the form $\mathrm{diag}(1, \ldots, 1, \lambda)$, where $\lambda \in k^{\times}$ as in Step 1.

Step 4: Use EC4 and EC6 when $\mathrm{char}(k)$ is odd or use EC8 and EC9 when $\mathrm{char}(k)$ is even to make E zero. Note that the relation ${}^{T}A_0\beta_0 A_0 + {}^{T}FE + {}^{T}EF = \beta_0$ shows that A_0 is invertible. Thus the relation ${}^{T}A_0\beta_0 Y + {}^{T}FB + {}^{T}ED = 0$ guarantees that Y becomes zero.

Step 5: Use ER2 to make B a zero matrix. Thus the matrix g reduces to $g = \mathrm{diag}\left(A_0, 1, \ldots, \lambda, 1, \ldots, \lambda^{-1}\right)$. Now if $\mathrm{char}(k)$ is odd, then go to Step 6; otherwise go to Step 7.

Step 6: Using the relation ${}^{T}A_0\beta_0 A_0 = \beta_0$, it is easy to check that A_0 has the form $\begin{pmatrix} t & -\epsilon s \\ s & t \end{pmatrix}$ or $\begin{pmatrix} t & \epsilon s \\ s & -t \end{pmatrix}$. If the determinant of A_0 is -1, multiply g by x_2 to get new g of the above form such that A_0 has determinant 1. Now using the elementary matrix $x_1(t, s)$, we can reduce g to $\mathrm{diag}\left(I_2, 1, \ldots, \lambda, 1, \ldots, \lambda^{-1}\right)$.

Step 7: Using elementary matrix x_{A_0}, we can reduce g to $\text{diag}\left(I_2, 1, \ldots, \lambda, 1, \ldots, \lambda^{-1}\right)$.

Lemma A.2 *Let k be a field of characteristics 2 and let* $g = \begin{pmatrix} A_0 & X & Y \\ E & A & B \\ F & 0 & D \end{pmatrix}$, *where* $A = \text{diag}(1, 1, \ldots, 1, \lambda)$, *be an element of* $O^-(2l, k)$ *then* $X = 0$.

Proof. Let $\{e_1, e_{-1}, e_2, \ldots, e_l, e_{-2}, \ldots, e_{-l}\}$ be the standard basis of the vector space V. Recall that for a column vector $x = (x_1, x_{-1}, x_2, \ldots, x_l, x_{-2}, \ldots, x_{-l})^t$, the action of the quadratic form Q is given by $Q(x) = \alpha\left(x_1^2 + x_{-1}^2\right) + x_1 x_{-1} + \ldots + x_l x_{-l}$, where $\alpha t^2 + t + \alpha$ is irreducible over $k[t]$. By definition, for any $g \in O^-(2l, k)$, we have $Q(g(x)) = Q(x)$ for all $x \in V$. Let $X = \begin{pmatrix} x_{11} \cdots x_{1(l-1)} \\ x_{21} \cdots x_{2(l-1)} \end{pmatrix}$ be a $2 \times (l-1)$ matrix. Computing $Q(g(e_i)) = Q(e_i)$ for all $2 \le i \le l$, we can see that $\alpha\left(x_{1i}^2 + x_{2i}^2\right) + x_{1i}x_{2i} = 0$. If $x_{2i} = 0$ then we can see that $x_{1i} = 0$. Suppose $x_{2i} \neq 0$ for some i, then we rewrite the equation by dividing it by x_{2i} as $\alpha\left(\frac{x_{1i}}{x_{2i}}\right)^2 + \frac{x_{1i}}{x_{2i}} + \alpha = 0$, which is a contradiction to the fact that $\alpha t^2 + t + \alpha$ is irreducible over $k[t]$. Thus, $x_{2i} = 0$ for all $2 \le i \le l$ and hence $X = 0$. •

A.5 Time complexity of the above algorithms

We establish that the worst-case time complexity of the above algorithm is $\mathcal{O}\left(l^3\right)$. We mostly count the number of field multiplications.

Step 1: We make A a diagonal matrix by row-column operations that has complexity $\mathcal{O}\left(l^3\right)$.

Step 2: In making both C and B zero matrix, we multiply two rows by a field element and additions. In the worst case, it has to be done $\mathcal{O}(l)$ times and done $\mathcal{O}\left(l^2\right)$ many times. So the complexity is $\mathcal{O}\left(l^3\right)$.

Step 3: In odd-orthogonal group and twisted orthogonal group, we clear X, Y, E, F, this clearly has complexity $\mathcal{O}\left(l^2\right)$.

Step 4: This step has only a few operations that is independent of l.

Then clearly, the time complexity of our algorithm is $\mathcal{O}\left(l^3\right)$.

We have implemented the above algorithms in Magma [25]. For details of that implementation along with performance analysis of our algorithm, we refer to Bhunia et al. ([24], Section 8).

Author details

Sushil Bhunia[1], Ayan Mahalanobis[2]*, Pralhad Shinde[2] and Anupam Singh[2]

1 IISER Mohali, India

2 IISER Pune, Pune, India

*Address all correspondence to: ayan.mahalanobis@gmail.com

IntechOpen

References

[1] Monico C, Maze G, Rosenthal J. A public key cryptosystem based on action by semigroups. In: Proceedings of IEEE International Symposium on Information Theory. 2002

[2] Monico C, Maze G, Rosenthal J. Public key cryptography based on semigroup actions. Advances in Mathematics of Communications. 2007; **1**(4):489-506

[3] Climent J-J, Navarro PR, Tortosa L. An extension of the noncommutative Bergman's ring with a large number of noninvertible elements. Applicable Algebra in Engineering, Communication and Computing. 2014;**25**(5):347-361

[4] Grigoriev D, Kojevnikov A, Nikolenko SJ. Algebraic cryptography: New constructions and their security against provable break. St. Petersburg Mathematical Journal. 2009;**20**(6): 937-953

[5] Roman'kov V. Two general schemes of algebraic cryptography. Groups-Complexity-Cryptology. 2019, to appear

[6] Mahalanobis A. A simple generalization of the ElGamal cryptosystem to non-abelian groups II. Communications in Algebra. 2012; **40**(9):3583-3596

[7] Mahalanobis A. The MOR cryptosystem and finite p-groups. In: Contemporary Mathematics. Vol. 633. AMS; 2015. pp. 81-95

[8] Mahalanobis A, Singh A. Gaussian elimination in split unitary groups with an application to public-key cryptography. Journal of Algebra Combinatorics Discrete Structures and Applications. 2017;**4**(3):247-260

[9] Paeng S-H, Ha K-C, Kim JH, Chee S, Park C. New public key cryptosystem using finite non-Abelian groups. In: Kilian J, editor. Crypto 2001. LNCS. Vol. 2139. Springer-Verlag; 2001. pp. 470-485

[10] Monico C. Cryptanalysis of matrix-based MOR system. Communications in Algebra. 2016;**44**:218-227

[11] Barbulescu R, Gaudry P, Joux A, Thome E. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In: Eurocrypt2014. 2014. pp. 1-16

[12] Hoffstein J, Pipher J, Silverman JH. An Introduction to Mathematical Cryptography. Springer; 2008

[13] Knus M-A, Merkurjev A, Rost M, Tignol J-P. The Book of Involutions (English Summary) with a Preface in French by J. Tits. Vol. 44. American Mathematical Society Colloquium Publications; 1998

[14] Dieudonne J. On the automorphisms of the classical groups with a supplement by Loo-Keng Hua. Memoirs of the American Mathematical Society. 1951

[15] Menezes AJ, Yi-Hong W. The discrete logarithm problem in GL (n, q). Ars Combinatoria. 1997;**47**:23-32

[16] Steinberg R. Generators of simple groups. Canadian Journal of Mathematics. 1962;**14**:277-283

[17] Ree R. On some simple groups defined by C. Chevalley. Transactions of the American Mathematical Society. 1957;**84**:392-400

[18] Alperin JL, Bell RB. Groups and Representations. New York: Springer-Verlag; 1995

[19] Brooksbank P. Constructive recognition of classical groups in their

natural representation. Journal of Symbolic Computation. 2003;**35**:195-239

[20] Brooksbank P. Fast constructive recognition of black-box unitary groups. LMS Journal of Computation and Mathematics. 2003;**6**:162-197

[21] Costi E. Constructive membership testing in classical groups [PhD thesis]. Queen Mary, Univ. of London; 2009

[22] Grove LC. Classical groups and geometric algebra. Vol. 39. American Mathematical Society, Graduate Studies in Mathematics; 2002

[23] Carter R. Simple groups of Lie type. In: Pure and Applied Mathematics. Vol. 28. John Wiley & Sons; 1972

[24] Bhunia S, Mahalanobis A, Shinde P, Singh A. Gaussian elimination in symplectic and orthogonal groups. In: Tech. Report. IISER Pune; 2017. Available from: https://arxiv.org/abs/1504.03794

[25] Bosma W, Cannon J, Playoust C. The magma algebra system. I. The user language. Journal of Symbolic Computation. 1997;**24**(3-4):235-265

[26] Steinberg R. Lectures on Chevalley Groups. University Lecture Series, Vol. 66, American Mathematical Society; 2016