Breaking Megrelishvili protocol using matrix diagonalization

Muhammad Arzaki¹, Danang Triantoro Murdiansyah², Satrio Adi Prabowo¹

¹Computing Laboratory, School of Computing, Telkom University, Indonesia
²High Performance Computing Laboratory, School of Computing, Telkom University, Indonesia

E-mail: {arzaki, danangtri}@telkomuniversity.ac.id, prabowosatrioadi@students.telkomuniversity.ac.id

Abstract. In this article we conduct a theoretical security analysis of Megrelishvili protocol—a linear algebra-based key agreement between two participants. We study the computational complexity of Megrelishvili vector-matrix problem (MVMP) as a mathematical problem that strongly relates to the security of Megrelishvili protocol. In particular, we investigate the asymptotic upper bounds for the running time and memory requirement of the MVMP that involves diagonalizable public matrix. Specifically, we devise a diagonalization method for solving the MVMP that is asymptotically faster than all of the previously existing algorithms. We also found an important counterintuitive result: the utilization of primitive matrix in Megrelishvili protocol makes the protocol more vulnerable to attacks.

1. Introduction

A reliable communication between two or more participants is an essential feature in modern cryptography. The protection of the communication channel can be attained in many ways, one of them is by constructing a mutual secret key for the rightful parties. The Diffie-Hellman key exchange (DHKE) protocol in [1] is one of the most commonly used technique for creating such key. It allows the participants to concur on a specific key via an open channel. DHKE has also been extensively used as a building block for many new protocols. One of them is the Megrelishvili protocol whose theoretical notion was first presented in [2].

Megrelishvili protocol is a linear algebra-based modification of the conventional DHKE. The mutual secret key in this protocol is a vector over a particular finite field. To construct the key, the protocol employs matrix exponentiation and left-multiplication of a matrix by a vector. These two linear algebra operations cause the security of Megrelishvili protocol is not directly associated with a specific discrete logarithm
problem in a particular group. Hence, Megrelishvili key exchange procedure presents an alternative solution for the creation of the common key between two participants.

The unique characteristics of Megrelishvili protocol has attracted numerous investigations to explore its mathematical and computational properties [3, 4, 5, 6, 7, 8, 9]. Despite these numerous studies, until recently the practical application of this protocol has not been widely implemented. Nevertheless—from theoretical viewpoint—the comprehensive theoretical security analysis of this protocol still needs to be carried out. In this paper we study the Megrelishvili vector-matrix problem (MVMP [7, 8, 9, 10]) as a mathematical problem that strongly correlates with the security of Megrelishvili protocol. Specifically, we focus our investigation regarding the computational complexity of the MVMP that involves diagonalizable public matrix.

We formulate efficient algorithms that solve the MVMP which involves diagonalizable public matrix in Section 4. We also prove that these algorithms are asymptotically faster than the other existing attack methods. Additionally, we also discuss some of the experimental results in Section 5 to support our theoretical analysis and to evaluate the practical efficiency of the proposed technique.

2. Finite Fields and Linear Algebra over Finite Fields

The working of Megrelishvili protocol relies heavily on mathematical theories of matrices and vector spaces over finite fields. We first briefly discuss some of our basic mathematical notations and terminologies throughout this paper. We use the standard mathematical notations as in [7, 8, 9, 10]. We write \( \mathbb{F}_q \) to represent the finite fields of \( q \) elements, \( \mathbb{F}_q^n \) to denote an \( n \)-dimensional vector space over \( \mathbb{F}_q \), and \( GL_n(q) \) to signify the collection of all nonsingular (invertible) matrices over \( \mathbb{F}_q \). We write vectors with bold lowercase letters (e.g. \( \mathbf{v} \)) and matrices using bold capital letters (e.g. \( \mathbf{M} \)). The vectors are considered and handled as row or column matrices. Furthermore, we treat the product of a \( 1 \times n \) vector \( \mathbf{v} \in \mathbb{F}_q^n \) and an \( n \times 1 \) vector \( \mathbf{w} \in \mathbb{F}_q^n \) (i.e., \( \mathbf{v} \mathbf{w} \)) as a scalar value in \( \mathbb{F}_q \).

Given a finite field \( \mathbb{F}_q \), the order of any nonzero element \( \beta \in \mathbb{F}_q \), denoted by \( |\beta| \), is defined as the least positive integer \( \ell \) such that \( \beta^\ell = 1 \). It is well-known that \( 1 \leq |\beta| \leq q - 1 \) and \( |\beta| \) divides \( q - 1 \) [11, 12, 13]. A nonzero element \( \alpha \in \mathbb{F}_q \) is called a primitive element if and only if \( |\alpha| = q - 1 \). This condition also implies that for any nonzero \( \beta \in \mathbb{F}_q \) there exists an integer \( t \in [1, q - 1] \) such that \( \alpha^t = \beta \). The problem of determining the exponent \( t \) that satisfies \( \alpha^t = \beta \) where \( \alpha \) is the primitive element of \( \mathbb{F}_q \) and \( \beta \) is a nonzero element in \( \mathbb{F}_q \) is called the discrete logarithm problem (DLP) in \( \mathbb{F}_q \). We refer the reader to [14] for more detailed discussion regarding this problem.

A polynomial \( f(x) \) of degree \( n \in \mathbb{N} \) over \( \mathbb{F}_q \) is an expression \( \sum_{i=0}^{n} \alpha_i x^i \) where \( \alpha_i \in \mathbb{F}_q \) for all \( 0 \leq i \leq n \). We denote the set of all polynomials over \( \mathbb{F}_q \) by \( \mathbb{F}_q[x] \). The degree of a nonzero polynomial \( f(x) \), denoted by \( \deg(f(x)) \), is defined as the greatest integer \( k \) such that the coefficient of \( x^k \) is nonzero. A polynomial \( f(x) \in \mathbb{F}_q[x] \) is reducible if and only if \( f(x) = g(x) h(x) \) for some \( g(x), h(x) \in \mathbb{F}_q[x] \) with \( 0 < \deg(g(x)) < \deg(f(x)) \); otherwise, it is irreducible. An irreducible polynomial \( f(x) \) is called a primitive polynomial of degree \( d \) over \( \mathbb{F}_q \) if it is the polynomial with the least positive degree such that \( f(\alpha) = 0 \) where \( \alpha \) is a primitive element of \( \mathbb{F}_{q^d} \).

It is well-known that any finite field must contain a prime power number of elements,
that is, for any field $\mathbb{F}_q$, we have $q = p^m$ for some positive prime $p$ and positive integer $m$. The finite field $\mathbb{F}_{p^m}$ can be constructed by choosing an irreducible polynomial $f(x) \in \mathbb{F}_p[x]$ of degree $m$ and constructing the quotient ring $\mathbb{F}_p[x] / (f(x))$, i.e., the ring of polynomials over $\mathbb{F}_p$ where the addition and multiplication are defined modulo $f(x)$ [11, 12, 13]. As a result, the finite field $\mathbb{F}_{p^m}$ is isomorphic to $\mathbb{F}_p / (f(x))$. In this paper, we consider both of these finite fields as identical fields.

For any nonsingular $n \times n$ matrix $M$ over $\mathbb{F}_q$, the (multiplicative) order of $M$, written by $\text{ord}(M)$, is defined as the smallest positive integer $s$ such that $M^s = I$, where $I$ is the $n \times n$ identity matrix. From [15], we have $1 \leq \text{ord}(M) \leq q^n - 1$. Furthermore, we call $M$ as a primitive matrix of size $n$ over $\mathbb{F}_q$ whenever $\text{ord}(M) = q^n - 1$. Given a square matrix $M$ of size $n$ over $\mathbb{F}_q$, an eigenvector of $M$ is a nonzero vector $v \in \mathbb{F}_q^n$ that satisfies $Mv = \lambda v$ for some scalar $\lambda$. The scalar $\lambda$ is called an eigenvalue of $M$ corresponding to the eigenvector $v$. Moreover, a scalar $\lambda$ is an eigenvalue of $M$ if and only if $\det(\lambda I - M) = 0$. The polynomial $\det M(x) = \det (xI - M)$ is called the characteristic polynomial of $M$. Therefore, we can find all eigenvalues of $M$ by solving the characteristic equation $\det M(x) = 0$ for $x$. However, this equation does not always have solutions in $\mathbb{F}_q$. When this condition happens, we need to extend our field of consideration to the splitting field of $\det M(x)$ over $\mathbb{F}_q$, that is, the smallest field extension of $\mathbb{F}_q$ over which $\det M(x)$ decomposes to linear factors [12, 13]. Suppose $\det M(x)$ is a monic polynomial of degree $n$ over $\mathbb{F}_q$ which can be factored as $\det M(x) = \prod_{i=1}^{n} (x - \lambda_i)$ where each $\lambda_i$ is a root of $\det M(x)$, then the splitting field of $\det M(x)$ over $\mathbb{F}_q$ is denoted by $\mathbb{K}(\lambda_1, \lambda_2, \ldots, \lambda_n)$. In this situation, elements of a splitting field can be treated as polynomials or vectors over $\mathbb{F}_q$.

An $n \times n$ matrix $M$ over $\mathbb{F}_q$ is said to be diagonalizable if there exists a nonsingular matrix $P$ such that $P^{-1}MP$ is a diagonal matrix. Suppose $\Lambda = P^{-1}MP$ is a diagonal matrix, then it is easy to show that the diagonal entries of $\Lambda$ are the eigenvalues of $M$. Moreover, if we write $\Lambda = \text{diag} (\lambda_1, \lambda_2, \ldots, \lambda_n)$ and $P = [p_1 | p_2 | \cdots | p_n]$ where each $p_i$ is a column vector, then each $p_i$ is the eigenvector of $M$ associated with eigenvalue $\lambda_i$ for all $1 \leq i \leq n$. Hence, an $n \times n$ matrix $M$ is diagonalizable if and only if it has $n$ linearly independent eigenvectors [16, Theorem 1.3.7]. This condition is equivalent to the statement that $M$ is diagonalizable if and only if the geometric multiplicity of each eigenvalue of $M$ is identical to its algebraic multiplicity. Furthermore, from [16, Theorem 1.3.9] we know that $M$ is diagonalizable if all of its eigenvalues are distinct.

3. The Standard Megrelishvili Protocol and Megrelishvili Vector-Matrix Problem (MVMMP)

The standard two-party Megrelishvili protocol is an example of linear algebra-based variant of the generic Diffie-Hellman key agreement. The theoretical notion of this protocol was first presented in [2]. It uses matrix exponentiation and left-multiplication of a matrix by a row vector that are performed over a specific finite field. Throughout this paper, we utilize the formal description of Megrelishvili protocol as explained in [6, 7, 8, 9]. The protocol differs from the typical DHKE because its security does not straightforwardly relate to a specific variation of the DLP in a finite group. For the discussion pertaining to the algorithm analysis of Megrelishvili protocol and its comparison to other variants of Diffie-Hellman protocol, the reader is referred to [7].
Unlike the conventional DHKE, the common confidential keys of the participants involved in Megrelishvili protocol is a (row) vector in $\mathbb{F}_q^n$. Suppose there are two participants, namely participant 1 and participant 2 who want to construct a common secret key using the standard Megrelishvili protocol. Initially, both participants need to agree on several public parameters which consists of a finite field $\mathbb{F}_q$, a (large) integer $n$, a nonzero (row) vector $v \in \mathbb{F}_q^n$, and an invertible matrix $M \in GL_n(q)$. Each participant $i$ then constructs their respective secret matrix by raising the matrix $M$ to a power of a secret integer $t_i$. The result of this exponentiation is a secret matrix $S_i$ which is exclusively known by participant $i$ itself. For the key exchange, each participant $i$ left-multiplies $S_i$ by the public nonzero vector $v$ and sends the result (i.e., $vS_i$) to another participant via an open channel. Finally, each participant $i$ retrieves the mutual secret vector by left-multiplying $S_i$ by the vector obtained from its counterpart. Observe that we have $S_1S_2 = M^{t_1}M^{t_2} = M^{t_2}M^{t_1} = S_2S_1$, and thus $vS_1S_2 = vS_2S_1$, which signifies that both participants possess identical secret vectors at the end of the exchange. We summarize this standard two-party Megrelishvili key exchange procedure in Table 1.

| Public parameters of Megrelishvili protocol |
|---------------------------------------------|
| A reliable third party picks and announces: a finite field $\mathbb{F}_q$, a (large) integer $n$, a nonzero (row) vector $v \in \mathbb{F}_q^n$, and a matrix $M \in GL_n(q)$. |

| Participant 1 | Participant 2 |
|---------------|---------------|
| **Secret matrices generation** | |
| Choose an integer $t_1$. | Choose an integer $t_2$. |
| Compute $S_1 = M^{t_1}$. | Compute $S_2 = M^{t_2}$. |
| **Exchange of vectors via an open channel** | |
| Compute $a_1 = vS_1$. | Compute $a_2 = vS_2$. |
| Send $a_1$ to participant 2. | Send $a_2$ to participant 1. |
| **Mutual key retrieval for each participant** | |
| Compute $a'_1 = a_2S_1$. | Compute $a'_2 = a_1S_2$. |
| The common secret key is $a'_1 = a'_2$. |

By observation, it is desirable if the values of $vM^t$ are all distinct for as many integer $t$ as possible. However, since for any $M \in GL_n(q)$ we have $\text{ord}(M) \leq q^n - 1$, then $vM^t$ has at most $q^n - 1$ different values in $\mathbb{F}_q^n$. There is an algorithm in [8] that provides a method for constructing a public matrix $M$ whose order is precisely $q^n - 1$. The public matrix $M$ is generated using the companion matrix of a primitive polynomial of degree $n$ over $\mathbb{F}_q$. In such condition, $M$ is also a primitive matrix of size $n$ over $\mathbb{F}_q$. Moreover, this algorithm also entails that the values of $vM^t$ are all distinct for every $0 \leq t < q^n - 1$ for all nonzero vectors $v$, which means that $vM^t$ can be any nonzero vector in $\mathbb{F}_q^n$ [8, Theorem 3].

The exchange of vectors in Megrelishvili protocol is performed over an open channel which is possibly being monitored by a wiretapper. In this step, participant $i$ sends the vector $a_i = vP_i = vM^{t_i}$ to its counterpart. Since the values of $a_i$, $v$, and $M$ are public, a
wiretapper may intercept the message $a_i$ and unravel the value of the secret integer $t_i$ by solving the equation $a_i = vM^t$ for $t_i$. Consequently—from theoretical standpoint—the security of Megrelishvili protocol relates to the hardness of computing the value $t$ from the equation $w = vM^t$, where $v, w \in \mathbb{F}_q^n$ and $M \in GL_n(q)$. This problem is called the Megrelishvili vector-matrix problem and it is formally described in Definition 1.

**Definition 1 ([7, 8, 9, 10])** The Megrelishvili vector-matrix problem (MVMP) is a linear algebra problem of determining an integer $t$ (if such integer exists) from the equation $w = vM^t$ where $v$ and $w$ are row vectors in $\mathbb{F}_q^n$ and $M \in GL_n(q)$.

The MVMP is similar to the well-known discrete logarithm problem that underlies the security of the conventional Diffie-Hellman protocol. However, it should be noted that the values on both sides of the equation in the MVMP are vectors, which are not elements of finite multiplicative group. From Table 1, a wiretapper can obtain the mutual secret $a_i$ from $a_1 = vM^t$ and recovering the value $t_1$ from $a_1 = vM^{t_1}$ and recovering the value $t_2$ from $a_2 = vM^{t_2}$. Then, the mutual vector can be retrieved simply by computing $vM^{t_1+t_2}$ which can be done in polynomial number of scalar operations in $\mathbb{F}_q$. From [7, Theorem 5], we know that the MVMP in $\mathbb{F}_q^n$ can be solved using exhaustive search (brute-force) approach in $O(T_{\text{mat}}(n) \cdot \text{ord}(M))$ number of scalar operations, where $T_{\text{mat}}(n)$ is the time complexity of the preferred matrix multiplication algorithm. Furthermore, the fact that $T_{\text{mat}}(n) = O(n^3)$ [17, Chapter 4] and $\text{ord}(M) \leq q^n - 1$ gives us an $O((n^3q^n)$ upper bound for the number of scalar operations required to solve the MVMP. This exhaustive search approach also gives an $O(n)$ space complexity bound for storing the vectors [9].

If the public matrix $M$ is constructed using Algorithm 2 in [8], then the values of $vM^t$ are all distinct for all integers $0 \leq t \leq q^n - 2$. In this case, brute-force attack is not the only admissible technique for solving the MVMP. Using the adaptation of Shanks's baby-step giant-step algorithm in [18], there exists collision algorithms which solve the MVMP in $\mathbb{F}_q^n$ using $O(n^3 \cdot q^{n/2} \cdot \log q)$ scalar operations [9]. In other words, these collision algorithms solve the MVMP in $\mathbb{F}_q^n$ faster than the brute-force method by a factor of $O((1/n \log q) \cdot q^{n/2})$. Additionally, the numerical experiment in [9] also indicates that the collision algorithms practically work faster than the exhaustive search attack, with the crossover points as low as $n = 3$ and $q = 2$. Despite their superior speed, both theoretical and experimental results show that the memory consumption for the collision algorithms in [9] are larger than those for the brute-force approach.

4. Diagonalization Methods for Breaking Megrelishvili Protocol

4.1. General diagonalization method for solving the MVMP

We first present a general approach for solving the MVMP using matrix diagonalization method. The idea of our technique is inspired by the research concerning the DLP in $GL_n(q)$ in [19, 20]. In both of these papers, the authors exploit Jordan canonical forms of matrices—which can be considered as generalized forms of diagonal

---

1 This quantity is obtained under the assumption that $T_{\text{mat}}(n) = O(n^3)$ and $\text{ord}(M) = q^n - 1$.

2 More precisely, the memory usage for the collision algorithms in [9] is larger than that for the brute-force attack by a factor of up to $O(q^{n/2})$. 

---
matrices. However, here we restrict our interest to a special case when the public matrix $M$ is diagonalizable.

Given a public matrix $M \in GL_n(q)$, a public vector $v \in \mathbb{F}_q^n$, and a known vector $w \in \mathbb{F}_q^n$ that is transmitted over an open channel, our objective is to find an integer $t$ (if such value exists) such that

$$vM^t = w. \quad (1)$$

If $M$ is diagonalizable, then we can write $M = P\Lambda P^{-1}$, where $\Lambda$ is a diagonal matrix whose diagonal entries are the eigenvalues of $M$ and $P$ is an invertible matrix whose columns correspond to the eigenvectors of $M$. Consequently, (1) becomes

$$v(P\Lambda P^{-1})^t = w \quad (2)$$

$$v(\Lambda^{-1}PA^{-1})^t = w \quad (3)$$

$$vPA^t = wP \quad (4)$$

To ease our subsequent analysis, we write $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$, where each $\lambda_i$ for $1 \leq i \leq n$ is the eigenvalue of $M$. Since $\Lambda$ is a diagonal matrix, from basic facts in matrix theory, we have $\Lambda^t = \text{diag}(\lambda_1^t, \lambda_2^t, \ldots, \lambda_n^t)$. Suppose we write $P = [p_1 \mid p_2 \mid \cdots \mid p_n]$ where each $p_i$ is the eigenvector of $M$ associated with eigenvalue $\lambda_i$ for $1 \leq i \leq n$. By considering the vectors $v$ and $w$ as $1 \times n$ row vectors and the vectors $p_i$ for all $1 \leq i \leq n$ as $n \times 1$ column vectors, we have

$$vP = v[p_1 \mid p_2 \mid \cdots \mid p_n] = [vp_1 \ vp_2 \ \cdots \ vp_n] \quad (5)$$

$$wP = w[p_1 \mid p_2 \mid \cdots \mid p_n] = [wp_1 \ wp_2 \ \cdots \ wp_n], \quad (6)$$

where each $vp_i$ and $wp_i$ are scalar values for all $1 \leq i \leq n$. For brevity, we write $vp_i = \hat{v}_i$ and $wp_i = \hat{w}_i$ for all $1 \leq i \leq n$ and

$$vP = [vp_1 \ vp_2 \ \cdots \ vp_n] = [\hat{v}_1 \ \hat{v}_2 \ \cdots \ \hat{v}_n] = \hat{v} \quad (7)$$

$$wP = [wp_1 \ wp_2 \ \cdots \ wp_n] = [\hat{w}_1 \ \hat{w}_2 \ \cdots \ \hat{w}_n] = \hat{w}. \quad (8)$$

Using the previously mentioned notations, (4) becomes

$$\hat{v}\Lambda^t = \hat{w} \quad (9)$$

$$[\hat{v}_1 \ \hat{v}_2 \ \cdots \ \hat{v}_n] \ \text{diag}(\lambda_1^t, \lambda_2^t, \ldots, \lambda_n^t) = [\hat{w}_1 \ \hat{w}_2 \ \cdots \ \hat{w}_n] \quad (10)$$

$$[\hat{v}_1 \lambda_1^t \ \hat{v}_2 \lambda_2^t \ \cdots \ \hat{v}_n \lambda_n^t] = [\hat{w}_1 \ \hat{w}_2 \ \cdots \ \hat{w}_n]. \quad (11)$$

All values $\hat{v}_i \lambda_i^t$ and $\hat{w}_i$ for all $1 \leq i \leq n$ are scalars in a particular extension field of $\mathbb{F}_q$. This extension field definitely must contain all eigenvalues $\lambda_i$ for all $1 \leq i \leq n$. Hence, it suffices to choose the extension field as the splitting field of $p_M(x)$ over $\mathbb{F}_q$, where $p_M(x)$ is the characteristic polynomial of $M$. We denote this splitting field by $\mathbb{K}(\lambda_1, \lambda_2, \ldots, \lambda_n)$. Clearly $\hat{v}_i \lambda_i^t$ and $\hat{w}_i$ for all $1 \leq i \leq n$ are elements of $\mathbb{K}(\lambda_1, \lambda_2, \ldots, \lambda_n)$. From the previous analysis, the values of $\hat{v}_i$, $\hat{w}_i$, and $\lambda_i$ can be identified by a wiretapper by determining the eigenvalues of the public matrix and their corresponding eigenvectors.

Since for all $1 \leq i \leq n$ the quantities $\hat{v}_i$ and $\hat{w}_i$ are elements of $\mathbb{K}(\lambda_1, \lambda_2, \ldots, \lambda_n)$, the quantities $\hat{v}_i \cdot (\hat{v}_i)^{-1}$ are well-defined and are scalars in $\mathbb{K}(\lambda_1, \lambda_2, \ldots, \lambda_n)$. Moreover,
because the finite field multiplication is commutative, we can represent \( \hat{w}_i \cdot (\hat{u}_i)^{-1} \) by \( \hat{w}_i / \hat{u}_i \) and accordingly denote this value by \( \hat{u}_i \). Therefore, from (11), we obtain

\[
\hat{u}_i \lambda_i^t = \hat{w}_i \text{ for all } 1 \leq i \leq n \tag{12}
\]

\[
\lambda_i^t = \hat{w}_i \cdot (\hat{u}_i)^{-1} = \hat{w}_i / \hat{u}_i = \hat{u}_i \text{ for all } 1 \leq i \leq n, \tag{13}
\]

where \( \hat{u}_i \) for all \( 1 \leq i \leq n \) are elements in the splitting field \( \mathbb{K}(\lambda_1, \lambda_2, \ldots, \lambda_n) \). Consequently, the integer \( t \) that satisfies (1) can be determined by solving \( n \) instances of the generic DLP in \( \mathbb{K}(\lambda_1, \lambda_2, \ldots, \lambda_n) \) of the forms \( \lambda_i^t = \hat{u}_i \) for all \( 1 \leq i \leq n \). In each of the DLP instances, every possible value of \( t \) is unique modulo \( |\lambda_i| \). Suppose the solutions of the \( n \) instances of the DLP are

\[
t \equiv t_1 \pmod{|\lambda_1|} \\
\vdots \\
\equiv t_n \pmod{|\lambda_n|} \tag{14}
\]

or in general \( t \equiv t_i \pmod{|\lambda_i|} \) with \( 1 \leq t_i \leq |\lambda_i| \) for all \( 1 \leq i \leq n \). The value \( t \) that satisfies (1) can be obtained by applying the generalized Chinese Remainder Theorem for non-coprime moduli to (14). From [21], we know that the solution to (14) exists if and only if \( t_i \equiv t_j \pmod{\lcm(|\lambda_1|, |\lambda_j|)} \) for all \( 1 \leq i < j \leq n \). Furthermore, if the solution exists, then it is unique modulo \( \lcm(|\lambda_1|, |\lambda_2|, \ldots, |\lambda_n|) \).

Suppose the characteristic polynomial of a public matrix \( \mathbf{M} \in GL_n(q) \) is a polynomial \( p_{\mathbf{M}}(x) \) over \( \mathbb{F}_q \) of degree \( n \). The aforesaid analysis shows that whenever the public matrix \( \mathbf{M} \) is diagonalizable, then the MVMP in \( \mathbb{F}_q^n \) can be reduced to at most \( n \) instances of the DLP in the splitting field of \( p_{\mathbf{M}}(x) \). From the facts in finite field theory, this splitting field is a subfield of the extension field \( \mathbb{F}_{q^n} \). Hence, whenever the public matrix is diagonalizable, the MVMP in \( \mathbb{F}_q^n \) is reduced to at most \( n \) instances of the DLP in \( \mathbb{F}_{q^n} \).

### 4.2. Efficient piecewise algorithm for solving the MVMP

The method in Section 4.1 stipulates that it is mathematically feasible to solve the MVMP using matrix diagonalization approach. However, mathematical theory alone is inadequate for asserting that matrix diagonalization method outperforms the other existing algorithms. From computational perspective, the existence of a rigorous and comprehensive algorithm that represents the approach in Section 4.1 is necessary. The previous mathematical analysis in Section 4.1 states that whenever the public matrix \( \mathbf{M} \in GL_n(q) \) is diagonalizable, the MVMP \( \mathbf{vM}^t = \mathbf{w} \) for \( \mathbf{v}, \mathbf{w} \in \mathbb{F}_q^n \) can be reduced to at most \( n \) instances of the DLP in the splitting field of the characteristic polynomial of \( \mathbf{M} \). This condition ostensibly requires the wiretapper to work with an enormous extension field of \( \mathbb{F}_q \) at once. In such circumstance, the algorithm would expect an immense time and storage requirement.

In this section we propose a rigorous and comprehensive piecewise method for solving the MVMP. Let \( \mathbf{M} \in GL_n(q) \) be a public matrix and \( p_{\mathbf{M}}(x) \in \mathbb{F}_q[x] \) be its characteristic polynomial. We assume that \( p_{\mathbf{M}}(x) = \prod_{i=1}^n (x - \lambda_i) \) where \( \lambda_1, \lambda_2, \ldots, \lambda_n \) are the roots (counting multiplicity) of \( p_{\mathbf{M}}(x) \). Instead of working with all roots of \( p_{\mathbf{M}}(x) \)
simultaneously, we may expect the diagonalization method to work more efficient if we handle each of the eigenvector one at a time. In such condition, we only need to consider the extension field \( \mathbb{K}(\lambda_i) \) whenever we deal with the eigenvalue \( \lambda_i \).

We describe the steps of our piecewise approach as follows. Initially, we need to determine the characteristic polynomial \( p_M(x) \) of the public matrix \( M \in GL_n(q) \). For obtaining \( p_M(x) \), the authors in [19] suggest Hessenberg algorithm—which requires polynomial number of scalar operations in \( \mathbb{F}_q \). Although other efficient algorithms do exist (see, e.g.: [22, 23]), it should be noted that the formation of \( p_M(x) \) for \( M \in GL_n(q) \) typically requires at most \( O(n^3) \) scalar operations in \( \mathbb{F}_q \).

The next step is the factorization of \( p_M(x) \) to linear factors which is required for determining the eigenvalues of \( M \). There are several techniques to perform polynomial factorization in \( \mathbb{F}_q[x] \). Three prominent methods are: Berlekamp algorithm, Cantor–Zassenhaus algorithm, and von zur Gathen–Shoup algorithm [24]. The complexities of these algorithms are measured in terms of the size of the finite field (denoted by \( q \)) and the degree of the polynomial (denoted by \( n \)). For sufficiently large values of \( q \) and \( n \), von zur Gathen–Shoup algorithm is the asymptotically fastest technique for factoring polynomials over \( \mathbb{F}_q \) of degree \( n \). This algorithm is expected to use \( O(n^2 \log^2 n \log \log n) \) number of scalar operations in \( \mathbb{F}_q \) [24]. In terms of memory consumption, Berlekamp algorithm requires \( O(n^2) \) space for storing the elements of \( \mathbb{F}_q \), while Cantor–Zassenhaus and von zur Gathen–Shoup algorithms respectively need \( O(n) \) and \( O(n^{3/2}) \) space for storing the elements of \( \mathbb{F}_q \).

After the factorization is completed, we assume that \( p_M(x) \) can be factored as \( \prod_{i=1}^{n}(x - \lambda_i) \) where \( \lambda_1, \lambda_2, \ldots, \lambda_n \) are the roots (counted with multiplicity). The subsequent important step is to check the diagonalizability of \( M \). This can be done by examining whether the geometric multiplicity of each eigenvalues of \( M \) is the same as its algebraic multiplicity. To do so, we first consider the distinct roots of \( p_M(x) \) and determine each of their algebraic multiplicities. For simplicity, the algebraic multiplicity of an eigenvalue \( \lambda \) is written by \( m_a(\lambda) \). If the roots of \( p_M(x) \) are all distinct, then \( M \) is diagonalizable. Otherwise, we need to check whether the geometric multiplicity of each eigenvalue \( \lambda \). Recall that the geometric multiplicity of an eigenvalue \( \lambda \), written by \( m_g(\lambda) \), is defined as the dimension of \( \ker(\lambda I - M) \).

Suppose \( M \) has \( r \) distinct eigenvalues \( \lambda_1, \lambda_2, \ldots, \lambda_r \) for some \( 1 \leq r \leq n \). The matrix \( M \) is diagonalizable if and only if \( m_a(\lambda_i) = m_g(\lambda_i) \) for every \( 1 \leq i \leq r \).

If the matrix \( M \) is diagonalizable, the next step is to determine \( n \) linearly independent eigenvectors associated with the eigenvalues of \( M \). To make our analysis easier, suppose the eigenvalues of \( M \), counted with multiplicity, are \( \lambda_1, \lambda_2, \ldots, \lambda_n \). For each \( \lambda_i \), we can find a nonzero vector \( \mathbf{p}_i \in \ker(\lambda_i I - M) \). Furthermore, the diagonalizability property of \( M \) ensures that there are \( n \) linearly independent vectors \( \mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n \) where each \( \mathbf{p}_i \) is associated with \( \lambda_i \).

The diagonalizability of \( M \) also implies that the quantities of algebraic multiplicities and geometric multiplicities of every eigenvalue are equal. Accordingly, we denote the algebraic multiplicity of eigenvalue \( \lambda_i \) by \( d_i \), which is also identical to the exponent of the factor \( (x - \lambda_i) \) within the factorization of \( p_M(x) \). To avoid working with all eigenvalues of \( M \) at once, we do not construct the matrix \( P \) whose columns are eigenvectors of \( M \). Instead, for every eigenvalue \( \lambda_i \) where \( 1 \leq i \leq n \), we work with the extension field
\( K(\lambda_i) = \mathbb{F}_q[x] / (x - \lambda_i)^{d_i} \cong \mathbb{F}^d_{q^{d_i}} \). Afterward, we consider the known vectors \( \mathbf{v} \) and \( \mathbf{w} \) as row vectors in \( \mathbb{F}^d_{q^{d_i}} \) and the eigenvector \( \mathbf{p}_i \) as column vector in \( \mathbb{F}^d_{q^{d_i}} \). Then, we compute \( \mathbf{v} \mathbf{p}_i \) and \( \mathbf{w} \mathbf{p}_i \) as scalar values in \( \mathbb{F}_{q^{d_i}} \) and accordingly denote these values by \( \hat{v}_i \) and \( \hat{w}_i \). Using (13), we calculate \( \hat{u}_i = \hat{w}_i / \hat{v}_i \) and obtain the DLP \( \lambda_i^t = \hat{u}_i \) in \( \mathbb{F}_{q^{d_i}} \). The value \( t \) that satisfies \( \lambda_i^t = \hat{u}_i \) in \( \mathbb{F}_{q^{d_i}} \) can be found using the algorithm for solving the DLP in a finite field (e.g.: [18, 25]). One of the commonly used nontrivial algorithms for solving the DLP in \( \mathbb{F}_q \) is the Shanks's baby step – giant step method, which solves the DLP in \( \mathbb{F}_{q^{d_i}} \) using \( O(d_i \cdot q^{d_i/2} \log q) \) scalar operations and \( O(q^{d_i/2}) \) space in \( \mathbb{F}_{q^{d_i}} \) [14].

Observe that with this piecewise approach, for every \( 1 \leq i \leq n \) we solve each of the DLP \( \lambda_i^t = \hat{u}_i \) in \( K(\lambda_i) \cong \mathbb{F}_{q^{d_i}} \) rather than in \( K(\lambda_1, \lambda_2, \ldots, \lambda_n) \). Assuming that the secret exponent \( t \) of the MVMP exists, the process of solving the DLP \( \lambda_i^t = \hat{u}_i \) in \( \mathbb{F}_{q^{d_i}} \) for every \( 1 \leq i \leq n \) yields an integer \( t_i \) that is unique modulo \( |\lambda_i| \). Because there are \( n \) instances of the DLP \( \lambda_i^t = \hat{u}_i \), the secret exponent \( t \) must satisfy \( t \equiv t_i \pmod{|\lambda_i|} \) with \( 1 \leq t_i \leq |\lambda_i| \) for all \( 1 \leq i \leq n \) as in (14). The value of \( t \) can be determined using the generalized Chinese Remainder Theorem for non-coprime moduli. Moreover, the solution \( t \) exists if and only if \( t_i \equiv t_j \pmod{\gcd(|\lambda_i|, |\lambda_j|)} \) for all \( 1 \leq i < j \leq n \). Additionally, if \( t \) exists, then \( t \) is unique modulo \( m = \text{lcm}(|\lambda_1|, |\lambda_2|, \ldots, |\lambda_n|) \). From [26], the complexity of this generalized Chinese Remainder Theorem is \( O(\log^2(m)) \).

The pseudocode for the efficient piecewise approach for solving the MVMP is depicted in Algorithm 1. The efficiency of this method is measured in terms of the total number of scalar operations in \( \mathbb{F}_q \) and the storage consumption under the assumption that one scalar value in \( \mathbb{F}_q \) requires \( O(1) \) space. Elements in \( \mathbb{F}_{q^{d_i}} \) are considered as \( d_i \) dimensional vectors or polynomials of degree \( d_i \) over \( \mathbb{F}_q \). Consequently, one scalar operation in \( \mathbb{F}_{q^{d_i}} \) is equivalent to \( d_i \) finite field operations in \( \mathbb{F}_q \) and one scalar value in \( \mathbb{F}_{q^{d_i}} \) needs \( O(d_i) \) space. The asymptotic running time and storage complexities of Algorithm 1 is stated in Theorem 1.

**Theorem 1** Assuming that Algorithm 1 uses Hessenberg procedure, von zur Gathen–Shoup factorization technique, and Shanks’s baby step – giant step method, the MVMP \( \mathbf{vM}^t = \mathbf{w} \) for any diagonalizable matrix \( \mathbf{M} \in GL_n(q) \) and \( \mathbf{v}, \mathbf{w} \in \mathbb{F}_q^n \) is solvable using \( O(n^3 \cdot q^{n^2} \cdot \log q) \) scalar operations and \( O(n^2 \cdot q^{n/2}) \) space in \( \mathbb{F}_q \).

*Proof:* We first analyze the asymptotic running time for Algorithm 1 as follows:

1. Line 1 is performed using Hessenberg algorithm which requires \( O(n^3) \) scalar operations in \( \mathbb{F}_q \).
2. Line 2 is the factorization of the characteristic polynomial of \( \mathbf{M} \). Using von zur Gathen–Shoup algorithm, this step requires \( O(n^2 \log^2 n \log \log n) \) operations in \( \mathbb{F}_q \).
3. Lines 3 and 5 involve arrays of length \( n \) whose entries are integers between 1 and \( n \) (inclusive). These lines are respectively performed by simple array scan and array matching procedures, both require at most \( O(n \log n) \) time. This is because if \( q \geq n \), then both array scan and array matching procedures require no more than \( O(n) \) time while if \( q < n \), then \( q^k = n \) for some \( k > 0 \), and thus both procedures require no more than \( O(n \log n) \) time.
Algorithm 1 Piecewise approach for solving the MVMP

Require: Vectors $\mathbf{v}, \mathbf{w} \in \mathbb{F}_q^n$ and a public matrix $\mathbf{M} \in GL_n(q)$.
1: Compute the characteristic polynomial of $\mathbf{M}$, i.e., $p_\mathbf{M}(x)$.
2: Factorize $p_\mathbf{M}(x)$ into $\prod_{i=1}^n (x - \lambda_i)$ using factorization algorithm.
3: For each $1 \leq i \leq n$, determine the value $d_i = m_a(\lambda_i)$ which is the algebraic multiplicity of $\lambda_i$ (i.e., the exponent of $(x - \lambda_i)$ within the factorization of $p_\mathbf{M}(x)$).
4: For each $1 \leq i \leq n$, determine the value $m_g(\lambda_i)$ which is the geometric multiplicity of $\lambda_i$ (i.e., $m_g(\lambda_i) = \dim (\ker (\lambda_i \mathbf{I} - \mathbf{M}))$).
5: For each $1 \leq i \leq n$, determine whether $m_a(\lambda_i) = m_a(\lambda_i)$. If their quantities are different, then $\mathbf{M}$ is not diagonalizable and the algorithm is terminated. Otherwise, $\mathbf{M}$ is diagonalizable and the algorithm is continued.
6: For all $1 \leq i \leq n$, find a nonzero vector $\mathbf{p}_i \in \mathbb{F}_q^{n+i}$ such that $\mathbf{p}_i \in \ker (\lambda_i \mathbf{I} - \mathbf{M})$ and the set $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n\}$ is linearly independent.
7: for all $i \in [1, n]$ do
8: Compute $\hat{v}_i \leftarrow \mathbf{v} \mathbf{p}_i$ as a value in $\mathbb{F}_q^{n+i}$. 
9: Compute $\hat{w}_i \leftarrow \mathbf{w} \mathbf{p}_i$ as a value in $\mathbb{F}_q^{n+i}$.
10: Compute $\hat{u}_i \leftarrow \hat{w}_i / \hat{v}_i$ as a value in $\mathbb{F}_q^{n+i}$.
11: Determine the exponent $t_i$ of the DLP $\lambda_i^{t_i} = \hat{u}_i$ in $\mathbb{F}_q^{n+i}$.
12: end for
13: Determine the solution of the systems of congruences $t \equiv t_i \pmod{|\lambda_i|}$ for $1 \leq i \leq n$ (as in (14)) using the generalized Chinese Remainder Theorem.
14: If the integer $t$ exists, then $t$ is the solution of $\mathbf{v} \mathbf{M}^t = \mathbf{w}$. Otherwise, the MVMP $\mathbf{v} \mathbf{M}^t = \mathbf{w}$ has no solution for $t$.

Ensure: If the algorithm is terminated and an integer $t$ exists, then $t$ is the solution of the MVMP $\mathbf{v} \mathbf{M}^t = \mathbf{w}$.

(4) Each of lines 4 and 6 can be done in $\mathcal{O}(Dn^4)$ time where $D = \max \{d_1, d_2, \ldots, d_n\}$. There are $n$ eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ and each $\lambda_i$ is an element of $\mathbb{F}_{q^{n+i}}$. The computation $\ker (\lambda_i \mathbf{I} - \mathbf{M})$ for each $1 \leq i \leq n$ is performed over $\mathbb{F}_{q^{n+i}}$ and it requires $\mathcal{O}(d_i n^3)$ scalar operations in $\mathbb{F}_q$. Since there are $n$ eigenvalues, the upper bound for each of lines 4 and 6 becomes $\sum_{i=1}^n \mathcal{O}(d_i n^3) = \sum_{i=1}^n \mathcal{O}(Dn^3) = \mathcal{O}(Dn^4)$.

(5) Each of lines 8 and 9 requires $\mathcal{O}(d_i n^2)$ scalar operations in $\mathbb{F}_q$. This is because the vectors $\mathbf{v}$, $\mathbf{w}$, and $\mathbf{p}_i$ are respectively considered as matrices over $\mathbb{F}_{q^{n+i}}$ of size $1 \times n$, $1 \times n$, and $n \times 1$.

(6) Line 10 requires $\mathcal{O}(d_i^2)$ operations in $\mathbb{F}_q$ because both $\hat{v}_i$ and $\hat{w}_i$ are considered as polynomials of degree $d_i$ over $\mathbb{F}_q$.

(7) Line 11 is performed using the Shanks’s baby step – giant step algorithm which requires $\mathcal{O}(d_i \cdot q^{d_i/2} \cdot \log q)$ operations in $\mathbb{F}_{q^{n+i}}$, or equivalently $\mathcal{O}(d_i^2 \cdot q^{d_i/2} \cdot \log q)$ operations in $\mathbb{F}_q$.

(8) Line 13 is the generalized Chinese Remainder Theorem for non-coprime moduli which requires $\mathcal{O}(\log^2 D) = \mathcal{O}(\log^2 n)$ time. This upper bound follows from the condition $D = \max \{d_1, d_2, \ldots, d_n\}$ and $d_i \leq n$ for all $1 \leq i \leq n$. 


From the previous analysis, the asymptotic time complexity for lines 8–11 is given by 
\[ O(d_i n^2) + O(d_i n^2) + O(d_i^2) + O(d_i^2 q^{d_i/2} \cdot \log q) \]. From the facts in finite field theory we have \( d_i \leq n \), which leads to the following upper bound for this asymptotic expression:
\[
O(n^3) + O(n^3) + O(n^2) + O\left(n^2 \cdot q^{n/2} \cdot \log q\right) = O\left(n^3 + n^2 \cdot q^{n/2} \cdot \log q\right)
\]
\[ = O\left(n^2 \cdot q^{n/2} \cdot \log q\right), \text{ because } q \geq 2. \tag{15}\]

Consequently, the upper bound for the asymptotic running time of the while loop (lines 7–12) is 
\[ n \cdot O\left(n^2 \cdot q^{n/2} \cdot \log q\right) = O\left(n^3 \cdot q^{n/2} \cdot \log q\right). \]
Therefore, the overall running time complexity of the algorithm can be analyzed as follows:

Table 2: Running time complexity of the algorithm

| line | time complexity |
|------|-----------------|
| 1    | \(O(n^3)\)      |
| 2    | \(O(n^2 \log^2 n \log \log n)\) |
| 3, 5 | \(O(n \log n)\) |
| 4, 6 | \(O(Dn^3) = O(n^5), \text{ since } D \leq n\) |
| 7–12 | \(O(n^3 \cdot q^{n/2} \cdot \log q)\) |
| 13   | \(O(\log^2 n)\) |
| overall | \(O(n^3 \cdot q^{n/2} \cdot \log q)\) |

since \(O(n^3 \cdot q^{n/2} \cdot \log q)\) dominates other terms.

For the space complexity analysis of Algorithm 1, observe that an element of \(\mathbb{F}_{q^{d_i}}\) can be viewed as a polynomial (or a vector) of degree (or dimension) \(d_i\) over \(\mathbb{F}_q\). To ease our analysis in determining the upper bound for the memory complexity, we may assume that the storage requirement of an element in \(\mathbb{F}_{q^{d_i}}\) is equivalent to the storage requirement of \(n\) elements in \(\mathbb{F}_q\) (because \(d_i \leq n\)). The asymptotic space complexity is analyzed as follows:

1. Line 1 requires \(O(n^2)\) space for storing a constant number of \(n \times n\) matrices over \(\mathbb{F}_q\).
2. Line 2 requires \(O(d_i n^{3/2}) = O(n^{5/2})\) space for the factorization of a polynomial of degree \(n\) where each coefficient takes no more than \(n\) possible values.
3. Each of lines 3 and 5 uses an array of length \(n\) whose entries are integers between 1 and \(n\) (inclusive). If \(q \geq n\), then both arrays require no more than \(O(n)\) space; otherwise \(q^k = n\) for some \(k > 0\) and thus the space requirement becomes \(O(n \log n)\). Therefore, each of these lines requires at most \(O(n \log n)\) space.
4. In each of lines 4 and 6, the algorithm uses \(n\) matrices of size \(n \times n\) over \(\mathbb{F}_{q^{d_i}}\), where each \(d_i \leq n\). Thus, each of these lines needs \(O(n^4)\) space.
5. Each of lines 8 and 9 utilizes an \(n\) dimensional vector over \(\mathbb{F}_{q^{d_i}}\). Since \(d_i \leq n\), each of these lines requires \(O(n^2)\) space.
(6) Line 10 involves the value $\hat{u}_i$, $\hat{v}_i$, and $\hat{w}_i$. The space complexity for this line is $O(n)$ because $\hat{u}_i$, $\hat{v}_i$, and $\hat{w}_i$ are elements of $\mathbb{F}_{q^d}$. and $d_i \leq n$.

(7) Line 11 involves the computation of Shanks’s baby step – giant step algorithm in $\mathbb{F}_{q^d}$, which requires $O(n \cdot q^{n/2})$ space. This is because the Shanks’s baby step – giant step procedure needs $O(d_i \cdot q^{d_i/2})$ space in $\mathbb{F}_{q^d}$ or equivalently $O(n \cdot q^{d_i/2})$ space in $\mathbb{F}_{q^d}$.

(8) Line 13 requires $O(n^2)$ space for storing an array of integers between 1 and $q^n - 1$ (inclusive) of length $n$. There are $n$ entries where each entry requires at most $O(n)$ space.

Based on the aforementioned analysis, the asymptotic space complexity of lines 8–11 is $O(n^2) + O(n^2) + O(n) + O(n^2) = O(n^2)$ since $q \geq 2$. Consequently, the upper bound for storage requirement of the while loop (line 7–12) becomes $O(n^2 \cdot q^{n/2})$. Thus, the overall space complexity of the algorithm can be evaluated as follows:

| line | space complexity |
|------|------------------|
| 1    | $O(n^2)$         |
| 2    | $O(n^{5/2})$     |
| 3, 5 | $O(n \log n)$   |
| 4, 6 | $O(n^4)$         |
| 7–12 | $O(n^2 \cdot q^{n/2})$ |
| 13   | $O(n^2)$         |

Thus, the overall space complexity of the algorithm can be evaluated as:

$$O(n^2 \cdot q^{n/2})$$

because $O(n^2 \cdot q^{n/2})$ dominates other terms. □

The result in Theorem 1 states that our proposed diagonalization method can solve the MVMP in $\mathbb{F}_q^n$ using $O(n^3 \cdot q^{n/2} \cdot \log q)$ scalar operations, which means that Algorithm 1 solves the MVMP for diagonalizable matrix faster than the collision algorithm in [9] by a factor of $O(n)$. Nevertheless, the diagonalization technique requires $O(n^2 \cdot q^{n/2})$ space in its execution, which is larger than that of the collision algorithm by a factor of $O(n)$. This theoretical result indicates a theoretical time-memory trade-off of Algorithm 1.

4.3. Optimized diagonalization method for primitive matrices

Superficially, it is naturally desirable if the public matrix $M \in GL_n(q)$ is constructed using primitive matrix as in [8, Algorithm 2]. This is because if $M$ is a primitive matrix, then the value of $\mathbf{v}M^t$ are all distinct for any nonzero $\mathbf{v} \in \mathbb{F}_q^n$ and any integer $0 \leq t < q^n - 1$, which implies that $\mathbf{v}M^t$ can be any nonzero vector in $\mathbb{F}_q^n$ [8, Theorem 3]. Consequently, the utilization of primitive matrix for the formation of $M$ undoubtedly thwarts exhaustive search attack for unraveling the secret exponent. However, in this section we show that primitive matrices are diagonalizable and thus are vulnerable to the
attack method described in Section 4.2. Moreover, we devise a computationally faster version of Algorithm 1 for solving the MVMP which involves primitive matrices.

If $M \in GL_n(q)$ is a primitive matrix, then its characteristic polynomial, $p_M(x)$, is a primitive polynomial of degree $n$ over $\mathbb{F}_q$ [12]. Moreover, $p_M(x)$ can be factored as $p_M(x) = \prod_{i=0}^{n-1} (x - \xi^q)$ where $\xi$ is a primitive root of $\mathbb{F}_q^n$ [11, 12]. In addition, the values of $\xi^q$ are different for all $0 \leq i \leq n-1$ and are also primitive elements of $\mathbb{F}_q^n$. This implies that $M$ is diagonalizable and the diagonal matrix $\Lambda$ in (9) has the form $\Lambda = \text{diag} \left\{ \xi, \xi^q, \ldots, \xi^{q^{n-1}} \right\}$. Assuming that $\lambda_i = \xi^q$ for $1 \leq i \leq n$, the expressions (12) and (13) respectively become

$$\hat{v}_i \left( \xi^{q^j} \right)^t = \hat{w}_i \text{ for all } 1 \leq i \leq n$$

$$\left( \xi^{q^j} \right)^t = \hat{w}_i \cdot (\hat{v})^{-1} = \hat{w}_i / \hat{v}_i = \hat{u}_i \text{ for all } 1 \leq i \leq n.$$  

(16)

(17)

For each $1 \leq i \leq n$, the integer $t$ that satisfies (17) can be found using the Shanks’s baby step – giant step algorithm.

Suppose $t_i$ where $1 \leq i \leq n$ is the solution of (17), then the solution of the MVMP exists if for all $1 \leq i \leq n$ the congruences $t \equiv t_i \mod \xi^{q^j}$ have a unique solution. Using the fact in [21], the value of $t$ exists if and only if $t_i \equiv t_j \mod \gcd \left( \left| \xi^{q^j} \right|, \left| \xi^{q^j} \right| \right)$ for all $1 \leq i < j \leq n$. Since each $\xi^q$ for every $1 \leq i \leq n$ is a primitive element in $\mathbb{F}_q^n$, then we have $\left| \xi^{q^j} \right| = q^n - 1$ for all $1 \leq i \leq n$. Hence, the solution of the MVMP exists if and only if $t_i \equiv t_j \mod q^n - 1$ for all $1 \leq i < j \leq n$. Accordingly, we have the following corollary.

**Corollary 1** Suppose $M \in GL_n(q)$ is a primitive matrix, $v, w \in \mathbb{F}_q^n$, and $\xi$ is an eigenvalue of $M$. The MVMP $vM^t = w$ is solvable for $t$ if and only if the DLP $\xi^t = \hat{u}_i$ in $\mathbb{F}_q^n \cong \mathbb{F}_q/(p_M(x))$ has a solution for $t \in [0, q^n - 2]$.

**Proof:** The condition $t_i \equiv t_j \mod q^n - 1$ for all $1 \leq i < j \leq n$ implies $t_i \equiv t_1 \mod q^n - 1$ for all $1 \leq i \leq n$. Hence, $t$ must satisfy $\xi^t = \hat{u}_i$. □

Corollary 1 stipulates that instead of solving $n$ instances of the DLP as in Section 4.2, the MVMP in $\mathbb{F}_q^n$ that involves primitive matrix can be solved by solely solving a specific instance of the DLP in $\mathbb{F}_q^n$. This condition also implies that we only need to determine a pair of eigenvalue and eigenvector of $M$ instead of $n$ pair of eigenvalues and eigenvector as in Section 4.2. The pseudocode for this efficient method is described in Algorithm 2. The asymptotic running time and memory complexities of this algorithm is briefly discussed in Theorem 2.

**Theorem 2** Assuming that Algorithm 2 uses Hessenberg procedure and Shanks’s baby step – giant step method, the MVMP $vM^t = w$ for any primitive matrix $M \in GL_n(q)$ and $v, w \in \mathbb{F}_q^n$ is solvable using $O \left( n^2 \cdot q^{n/2} \cdot \log q \right)$ scalar operations and $O \left( n \cdot q^{n/2} \right)$ space in $\mathbb{F}_q$.
Algorithm 2 Optimized diagonalization approach to solve the MVMP for primitive matrices

Require: Vectors $v, w \in \mathbb{F}_q^n$ and a public matrix $M \in GL_n(q)$ which is a primitive matrix.
1: Compute the characteristic polynomial of $M$, i.e., $p_M(x)$.
2: Set the finite field $\mathbb{F}_{q^n} \cong \mathbb{F}_q / (p_M(x))$.
3: Find a nonzero vector $p \in \mathbb{F}_{q^n}$ such that $p \in \ker(\xi I - M)$.
4: Compute $\hat{v} \leftarrow vp$ as a value in $\mathbb{F}_{q^n}$.
5: Compute $\hat{w} \leftarrow wp$ as a value in $\mathbb{F}_{q^n}$.
6: Determine the exponent $t$ of the DLP $\xi^t = \hat{u}$ in $\mathbb{F}_{q^n}$.

Ensure: If the value $t$ exists, then $t$ is the solution of the MVMP $vM^t = w$.

Proof: The asymptotic analysis of this algorithm is almost similar to that described in the proof of Theorem 1. One substantial difference is that Algorithm 2 only considers an eigenvalue and one of its associated eigenvector instead of $n$ eigenvalues and their corresponding eigenvectors as in Algorithm 1. Hence, both asymptotic time and space complexities of Algorithm 2 are lower by a factor of $O(n)$ than those for Algorithm 1. Therefore, the running time and space complexities of Algorithm 2 are respectively $O(n^2 \cdot q^{n/2} \cdot \log q)$ and $O(n \cdot q^{n/2})$. The detail of the proof of this theorem is left to readers.

From the result in Theorem 2, we infer that Algorithm 2 is asymptotically faster than the collision algorithms in [9] by a factor of $O(n^2)$. Additionally, the space requirement of Algorithm 2 is asymptotically equivalent to those of the collision algorithms in [9]. Hence, from theoretical perspective, Algorithm 2 asymptotically outperforms the collision algorithms in [9].

5. Example and Numerical Experiments

In this section we discuss an example and numerical experiments regarding the formerly explained diagonalization technique for solving the MVMP. We first demonstrate a small numerical implementation of Algorithm 2 in Example 1.

Example 1 Suppose we consider the MVMP $vM^t = w$ in $\mathbb{F}_7^3$ with $v = (1, 1, 0)$, $w = (2, 0, 0)$, and $M = \begin{bmatrix} 4 & 4 & 0 \\ 4 & 5 & 5 \\ 0 & 4 & 2 \end{bmatrix}$ as in [9, Example 1]. The characteristic polynomial of $M$ is $p_M(x) = x^3 + 3x^2 + 2x + 2$ and it is a primitive polynomial of degree 3 over $\mathbb{F}_7$. Thus, we can apply Algorithm 2 to determine the secret exponent $t$ of the MVMP. Suppose $\xi$ is a root of $p_M(x)$ which is also an eigenvalue of $M$. The eigenvalue $\xi$ is associated with the eigenvector $p = (1, 2\xi + 6, 6\xi^2 + 2\xi + 3)^T$. Consequently, we have $\hat{v} = vp = 2\xi$ and $\hat{w} = wp = 2$, and thus the solution of the MVMP $vM^t = w$ is identical.
to the secret exponent $t$ in the following condition:

$$2\xi \left( \xi^t \right) = 2, \text{ or equivalently}$$

$$\xi^t = 2/2\xi = 3\xi^2 + 2\xi + 6. \quad (19)$$

By solving the DLP (19) in $\mathbb{F}_{7^3}$ for the value of $t$, we have $t = 341$.

The numerical experiments are performed to assess the practical effectiveness of the diagonalization method for solving the MVMP. We confine our subject of interest to the MVMP that uses primitive matrix as one of its public parameters. One underlying reason is because the utilization of primitive matrix purportedly makes the protocol more secure against exhaustive search attacks. The experiments were conducted by comparing both running time and memory aspects of the deterministic collision algorithm in [9] and Algorithm 2. The algorithms were implemented using SageMath version 7.1 beta3 and Python 2.7.17 over Mac OS X El Capitan system with 3.5 GHz 6-Core Intel Xeon E5 processor and 16 gigabytes of memory.

The experiments considered several instances of the MVMP in $\mathbb{F}_{q^n}$ for $q = 2, 3, 5, 7$ and various values of $n$. For each of the scenarios, the public matrix $M \in GL_n(q)$ is a primitive matrix, the public vector $v \in \mathbb{F}_{q^n}$ is a random nonzero vector, and the value of the secret exponent $t$ is equal to $\left\lfloor (q^n - 1)/2 \right\rfloor$. One objective of these experiments is to find the crossover points at which Algorithm 2 practically works faster than the deterministic collision algorithm. The graphs of the running time and storage requirement of both algorithms for solving the MVMP in $\mathbb{F}_{2^n}$ where $3 \leq n \leq 21$ are correspondingly depicted in Figure 1 and Figure 2. For the finite field $\mathbb{F}_2$, our experiments infer a crossover point of $n = 17$. This result shows that Algorithm 2 is practically faster than the collision algorithm in [9]. However, it requires more memory than the collision algorithm. We can see the memory usage comparison between Algorithm 2 and the collision algorithm in Figure 2. Their memory consumptions have constant difference around 9 megabytes. This disparity is constant because both algorithms have equivalent space complexity. Nevertheless, in our experiments, our proposed algorithm requires more memory than the collision algorithm. This is probably due to the utilization of extension fields in SageMath—which costs more space than the prime fields.

Table 4 presents the running time and memory consumption of the collision algorithm and Algorithm 2 for $\mathbb{F}_{q^n}$, where $q = 3, 5, 7$ and various values of $n$. Table 4 shows a similar property as in Figure 1 that Algorithm 2 is faster than the collision algorithm in [9]. The crossover points for these experiments are as low as $n = 17$ for $q = 2$, $n = 11$ for $q = 3$, $n = 7$ for $q = 5$, and $n = 7$ for $q = 7$. In addition, Table 4 also indicates similar characteristics as in Figure 2 that Algorithm 2 requires more memory than the collision algorithm. However, in our experiments, there are several anomalies. The memory consumption of our proposed algorithm when the matrix size is 14, 15, and 16 are higher than those other values in $\{3, 4, ..., 21\}$. Despite our best efforts, we do not know yet the explanation of these anomalies. We are currently planning to resolve this issue by conducting further experiments with more comprehensive test cases. Nevertheless, one should note that our experimental results are supplementary. Theoretically, our proposed algorithm uses asymptotically equivalent amount of memory to that of the collision algorithm as described in Theorem 2.
Figure 1: Average running time for collision algorithm in [9] and optimized diagonalization algorithm (Algorithm 2) for solving the MVMP in $\mathbb{F}_q^n$ for $q = 2$ and $n = 3, \ldots, 21$.

Figure 2: Average memory usage for collision algorithm in [9] and optimized diagonalization algorithm (Algorithm 2) for solving the MVMP in $\mathbb{F}_q^n$ for $q = 2$ and $n = 3, \ldots, 21$.

6. Conclusion and Future Works

We study the Megrelishvili vector-matrix problem (MVMP) as a linear algebra problem that strongly correlates with the security of Megrelishvili protocol. In particular, we formulate an approach for solving the MVMP using matrix diagonalization method. The theoretical result in Theorem 1 asserts that the MVMP $vM^t = w$ in $\mathbb{F}_q^n$ for any diagonalizable $M \in GL_n(q)$ is solvable using $O\left(n^3 \cdot q^{n/2} \cdot \log q\right)$ finite field operations and $O\left(n^2 \cdot q^{n/2}\right)$ storage. Moreover, our investigation also yields an important counterintuitive result: the utilization of a primitive matrix for the construction of the public matrix $M$ makes the protocol more vulnerable to attacks. In Theorem 2 we show that the MVMP in $\mathbb{F}_q^n$ which uses a primitive matrix is solvable in $O\left(n^2 \cdot q^{n/2} \cdot \log q\right)$ time and $O\left(n \cdot q^{n/2}\right)$ space. Therefore, our diagonalization methods are asymptotically faster than the collision algorithms in [9] whenever the public matrix is diagonalizable.
Table 4: Average running time and memory consumption for collision algorithm in [9] and optimized diagonalization method (Algorithm 2) for solving the MVMP in $\mathbb{F}_q^n$ for $q = 3, 5, 7$ and several $n$. 

| $q$ | $n$ | Collision Algorithm in [9] | Algorithm 2 |
|-----|-----|----------------------------|--------------|
|     |     | Running Time (seconds) | Memory Usage (megabytes) | Running Time (seconds) | Memory Usage (megabytes) |
| 3   | 11  | 0.54 | 1.1 | 0.46 | 10.53 |
| 12  | 0.81 | 1.17 | 0.51 | 10.53 |
| 13  | 1.35 | 1.47 | 0.64 | 10.5 |
| 5   | 7   | 0.40 | 1 | 0.39 | 10.2 |
| 8   | 0.63 | 1.13 | 0.48 | 10.23 |
| 9   | 1.25 | 1.13 | 0.63 | 10.73 |
| 7   | 6   | 0.46 | 0.93 | 0.47 | 10.2 |
| 7   | 0.9  | 1.23 | 0.54 | 10.4 |
| 8   | 2.23 | 2.13 | 0.87 | 10.8 |

Additionally, if the public matrix is also a primitive matrix, then diagonalization method asymptotically outperforms the collision algorithm in [9].

We also conducted numerical experiments to support the theoretical asymptotic analysis in Section 4. The experiments were restricted to the MVMP that uses primitive matrix as its public matrix. The results of the experiments show that the diagonalization method practically works faster than the deterministic collision algorithm for solving the MVMP in $\mathbb{F}_q^n$, with crossover points as low as $n = 17$ for $q = 2, n = 11$ for $q = 3, n = 7$ for $q = 5$, and $n = 7$ for $q = 7$. Nevertheless, experimental results indicate that the diagonalization method requires more memory than the collision algorithm—albeit the theoretical analysis shows that the space complexities of both techniques are asymptotically equivalent whenever the public matrix is also a primitive matrix.

In Theorem 1 and Theorem 2 we provide asymptotic running time upper bounds for solving the MVMP. These upper bounds improve the previously discovered bounds in [9]. However, these new upper bounds are confined to the condition where the public matrix is diagonalizable. We conjecture that a similar and more general upper bound for any public matrix can be obtained by applying the Jordan decomposition technique for solving the MVMP. We believe that this approach is attainable by modifying methods in [19, 20]. Finally, we do not know yet whether the upper bound for solving the MVMP that involves primitive matrices in Theorem 2 is optimal or not. The research of the optimal running time upper bound for solving the MVMP is essential to quantify the security of Megrelishvili protocol.

References
[1] Diffie W and Hellman M E 1976 New directions in cryptography, Information Theory, IEEE Transactions on 22 644–654
[2] Megrelishvili R, Chelidze M and Chelidze K 2006 New matrix sets generation and the cryptosystems, Applied Mathematics, Informatics and Mechanics (AMIM) 11 29–36
[3] Megrelishvili R and Sikharulidze A 2009 New matrix sets generation and the cryptosystems, Proceedings of the European Computing Conference and the Third International Conference on Computational Intelligence, Tbilisi, Georgia pp 253–255

[4] Megrelishvili R, Chelidze M and Besiashvili G 2010 Investigation of new matrix-key function for the public cryptosystems, Proceedings of The Third International Conference, Problems of Cybernetics and Information vol 1 pp 6–8

[5] Megrelishvili R 2011 On the original one-way function and the new digital signature scheme, Applied Mathematics, Informatics and Mechanics (AMIM) 16 43–48

[6] Beletsky A, Beletsky A and Kandyba R 2013 Matrix analogues of the Diffie-Hellman protocol, ICTERI pp 352–359

[7] Arzaki M 2016 Elementary algorithms analysis of Megrelishvili protocol, Indonesian Journal on Computing (Indo-JC) 1 11–23

[8] Arzaki M and Wahyudi B A 2016 On the construction of secure public parameters for Megrelishvili protocol, 2016 4th International Conference on Information and Communication Technology (ICoICT 2016) (IEEE) pp 1–5

[9] Arzaki M and Wahyudi B A 2016 Collision algorithms for breaking Megrelishvili protocol: theory and numerical experiments, 2016 8th International Conference on Advanced Computer Science and Information Systems (ICACISIS 2016) (IEEE) pp 53–60

[10] Arzaki M and Wahyudi B A 2017 Extending Megrelishvili protocol for multi-party key agreement, 2017 5th International Conference on Information and Communication Technology (ICoICT) (IEEE) pp 290–297

[11] McEliece R J 1987 Finite fields for computer scientists and engineers (Springer Science & Business Media)

[12] Lidl R and Niederreiter H 1994 Introduction to finite fields and their applications (Cambridge university press)

[13] Durbin J R 2008 Modern algebra: An introduction (John Wiley & Sons)

[14] Hoffstein J, Pipher J C and Silverman J H 2014 An introduction to mathematical cryptography 2nd ed (Springer)

[15] Niven I 1948 Fermat’s theorem for matrices, Duke Mathematical Journal 15 823–826

[16] Horn R A and Johnson C R 2012 Matrix analysis (Cambridge university press)

[17] Cormen T H, Leiserson C E, Rivest R E and Stein C 2009 Introduction to algorithms (MIT press)

[18] Shanks D 1971 Class number, a theory of factorization, and genera, Proc. Symp. Pure Math vol 20 pp 415–440

[19] Menezes A J and Wu Y H 1997 The discrete logarithm problem in $GL(n, q)$, Ars Combinatoria 47 23–32

[20] Freeman D M 2004 The discrete logarithm problem in matrix groups URL http://theory.stanford.edu/~dfreeman/papers/discretelogs.pdf

[21] Cohen H 2013 A course in computational algebraic number theory vol 138 (Springer Science & Business Media)

[22] Abdeljaoued J and Malaschonok G I 2001 Efficient algorithms for computing the characteristic polynomial in a domain, Journal of Pure and Applied Algebra 156 127–145

[23] Pernet C and Storjohann A 2007 Faster algorithms for the characteristic polynomial, Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation (ACM) pp 307–314

[24] Shoup V 1993 Factoring polynomials over finite fields: asymptotic complexity vs. reality, Proc. IMACS Symposium, Lille, France

[25] Pollard J M 1978 Monte Carlo methods for index computation (modp), Mathematics of Computation 32 918–924

[26] Shoup V 2009 A computational introduction to number theory and algebra (A computational introduction to number theory and algebra, Cambridge university press)