Algebraic Analysis of Object-Based Key Assignment Schemes

Khair Eddin Sabri
Computer Science Department
King Abdullah II School for Information Technology
The University of Jordan, Amman, Jordan
Email: k.sabri@ju.edu.jo

Abstract—The confidentiality of information is an important aspect of security. One way to achieve the confidentiality is through restricting access to information to the authorized users only. Access control can be enforced by using encryption. In this case, all the information is encrypted and keys are assigned to users such that each key reveals the authorized part of the information. Key assignment can be classified as key-based or object-based schemes based on the focus of the scheme.

Sabri and Khedri [1] present algebraic structures to specify algebraically cryptosystems by capturing the common properties of ciphers, secrets and keys. Also, these structures are used for the analysis of security properties in object-based key assignment schemes. However, these structures are abstract, and no linkage has been proposed to the existing cryptosystems. In this paper, we extend their work by giving concrete models for their algebraic structures. We give concrete models to Vigenère, transposition ciphers, DES and RSA cryptosystems. Also, we investigate the effects of extra algebraic properties that some cryptosystems may have on the security analysis of object-based schemes.

Index Terms—Object-based key assignment schemes, verification, cryptography, algebraic structures.

I. INTRODUCTION

Access control is used to provide confidentiality of information. The typical implementation of access control is through the use of trusted server that gives authorization to users based on predefined policies as in [2], [3], [4]. Another implementation that eliminates the use of a trusted server is through the use of cryptography such that the information is public but encrypted. Therefore, keys are assigned to users such that each key is used to decrypt the authorized part of the information.

Key assignment schemes can be classified into key-based schemes and object-based schemes [1]. Key-based schemes focus on keys and the relationship between them. It is usually used when we have hierarchy in the security labels assigned to users. For example, the relation $u_1 > u_2$ indicates that the user $u_1$ has more authority than the user $u_2$. In this case, the key $k_1$ assigned to user $u_1$ should be able to reveal more information than the key $k_2$ assigned to user $u_2$. We write $k_1 > k_2$ when any information that can be revealed using the key $k_2$ can also be revealed using the key $k_1$. One way to implement this relation is through deriving the key $k_2$ from the key $k_1$.

Several techniques are proposed in the literature to derive a key from another one as in [5], [6]. In [7], we introduce an algebraic model that links several techniques together to specify and analyze key-based schemes.

Object-based schemes focus on objects and the required condition to reveal the information from each object. This scheme is used when several users should cooperate to reveal an information. For example, conditions can be stated as the use of the keys $k_1$ and $k_2$ together to reveal a piece of information, while any one of the key $k_1$ or $k_2$ should be used to reveal another piece of information.

Sabri and Khedri [1] introduce algebraic structures to specify cryptosystems by specifying the common properties of secrets, ciphers, and keys. Also, they introduce an algebraic structure to specify the interaction between keys and ciphers and another algebraic structure to specify the process of encryption and decryption. They use their algebraic structures to verify, at an abstract level, security properties in systems that employed object-based schemes. Their analysis is based on the common algebraic properties of secrets, keys, and ciphers. However, the use of a specific cryptosystem may affect the satisfaction of security properties due to the fact that some cryptosystems have additional algebraic properties. For example, the composition of encryption of Vigenère is commutative $E_{k_1}(E_{k_2}(M)) = E_{k_2}(E_{k_1}(M))$ i.e., the order of multiple encrypting does not matter. Others such as RSA has the homomorphic property $E(M_1) \cdot E(M_2) = E(M_1 \cdot M_2)$ i.e., concatenating two encrypted messages is the same as encrypting after concatenating the two messages. The effects of the algebraic properties of cryptosystems are taken into consideration mainly in the verification of cryptographic protocols properties [8], [9], [10], where several flows are detected by involving the algebraic properties in the analysis other than the cancelling property $E_k(E_k(m)) = m$ i.e., the decryption cancels the effect of encryption.

In this paper, we extend the work of Sabri and Khedri [1] by providing concrete algebraic models of cryptosystems to their abstract structures. These cryptosystems are Vigenère, transposition cipher, DES and RSA [11]. We choose these cryptosystems because of...
their popularity and their algebraic properties that are useful to illustrate our idea. We are aware that Vigenère and transposition cryptosystems are not used in practice. However, we are presenting them in this paper because they are simple and easy to understand. Therefore, they are suitable to illustrate the algebraic structures presented in [1].

The presented concrete algebraic models act as an intermediate model between the abstract level of an algebraic structure and its implementation. Providing concrete models to abstract algebraic structures is not new. For example, Khedri [12] introduce a concrete model represented as relation algebra to an abstract algebraic model called feature algebra.

We also analyse the effect of the additional algebraic properties of the specified cryptosystems on the security properties of object-based key assignment schemes. To the best of our knowledge, security analysis of object-based schemes based on the algebraic properties of cryptosystems does not exist in the literature. It is worth mentioning that this paper does not investigate the security of cryptosystems themselves, but it analyse the effect of the algebraic properties of cryptosystems on the verification of object-based schemes.

This paper is organized as follows: Section II summarizes related work. Section III gives the necessary mathematical background and present the algebraic structures of [1]. Section IV analyses security properties in object-based schemes. Section V represents algebraically Vigenère, transposition cipher, DES and RSA cryptosystems. Finally, we conclude in Section VI and point to future works.

II. LITERATURE REVIEW

The algebraic properties of cryptosystems is used in many security applications. In this section, we present some of them. Even our algebraic model can be used in all these applications, the focus of this paper is on the analysis of key assignment schemes.

Key Assignment Schemes: Several techniques exist in the literature to provide confidentiality of information through the use of cryptography. Many of these techniques are key-based schemes [5], [13], [6], [14], [15]. These techniques provide a way to implement the relation \( k_1 > k_2 \) through key derivation i.e., deriving the key \( k_2 \) from the key \( k_1 \). Other researchers focus on analysing these techniques and comparing them [16], [7]. Others use key derivation to provide confidentiality in data outsourcing [17], [18].

A technique that follows the object-based scheme is the work of Miklau and Suciu [19], where the authors develop a language for specifying access control policies on XML. They represent policies as a tree where its nodes represent objects. Each object is associated with a condition that represents the required keys to reveal that object i.e., the use of the keys \( k_1 \) and \( k_2 \) together or the use of one of the keys \( k_1 \) or \( k_2 \). This tree is implemented by using the \( \text{xor} \) operator for secret sharing and AES cryptosystem for encryption. Miklau and Suciu scheme is analysed by Abadi and Warinschi [20]. In our paper, we specify object-based scheme similar to [19]. However, in this paper, we propose several implementations for object-based schemes and analyse the effect of each implementation on the security of schemes based on its algebraic properties.

Analysis of Cryptographic Protocols: Cryptographic protocols are communicated protocols whose messages are encrypted. They provide security properties such as confidentiality of information and authentication between communicated users. Several techniques are presented in the literature [21], [22], [23] to analyse these protocols and verify the satisfactory of their security properties. The analysis is based on abstracting the used cryptosystem in the analysed protocol by focusing on its algebraic properties. The main considered property is the cancellation property which states that encryption cancels the effect of encryption when appropriate keys are used. Researchers report that some cryptosystems have additional properties such as the commutativity and associativity of encryption. Using a cipher with such properties in some protocols would make those protocols vulnerable to attacks [10]. Therefore, new researches focus more on the algebraic properties that cryptosystems may have when analysing cryptographic protocols [24], [9], [25].

Threshold Cryptosystem: Secret sharing is an approach used for splitting the secret into shares and distributing them between \( n \) users such that for \( k \) of them should combine their shares to construct the secret. One of the earliest methods is the Shamir method [26] based on interpolation. This approach requires the exchange of the shares usually with a trusted party. Another approach that does not require exchanging the shares but has the same goal of revealing a secret is called threshold cryptosystem. In this approach, \( k \) users cooperate in the decryption of a cipher text using their keys to reveal the secret. One of the earliest research on RSA threshold is the work of Boyd [27] based on the algebraic property \( E_{k_2}(E_{k_1}(s)) = E_{k_1 \cdot k_2}(s) \). This property states that encryption/decryption of a secret twice using the keys \( k_1 \) and \( k_2 \) is equivalent to encrypting/decrypting the secret using a combined key \( k_1 \ast k_2 \).

Cryptanalysis: Cryptanalysis is the process of revealing a secret from an encrypted message without using a key. This field is intensively studied to analyse the security of cryptosystem. Representing cryptosystems algebraically presents another view of analysing them. This representation is used in [28], [29] to algebraically analyse the security of some cryptosystems such as Advanced Encryption Standard (AES).

III. MATHEMATICAL BACKGROUND

In this section, we first introduce the required mathematical background [30]. Then, we present the algebraic structures given by [1] to specify the elements of a cryptosystem and the encryption and decryption of secrets.
A. Algebraic Structures

Definition 3.1: A semigroup is an algebraic structure $A = (S, \cdot)$, where $S$ is a set and $\cdot$ is an associative binary operator. If the operator $\cdot$ is commutative, we call $A$ a commutative semigroup. If the operator $\cdot$ is idempotent, we call $A$ an idempotent commutative semigroup.

Definition 3.2: A group is an algebraic structure $G = (G, \cdot, 1)$, where $(G, \cdot)$ is a semigroup, $1$ is an identity element of $\cdot$, and for every element $g$ in $G$ there is an inverse element $g^{-1}$ such that $g \cdot g^{-1} = 1$.

Definition 3.3: Let $S \neq \emptyset$ be a set and $\cdot$ and $+$ be binary operations on $S$, named addition and multiplication. Then $(S, \cdot, +)$ is called a semiring if $(S, +)$ is a commutative semigroup, $(S, \cdot)$ is a semigroup, and distributes over $+$ on both the left and right.

Definition 3.4: Let $(S, +)$ be a semiring.

- If the semigroup $(S, \cdot)$ has a neutral element $1$, we call $1$ the identity of the semiring $(S, +, \cdot)$.
- If the semigroup $(S, +)$ has a neutral element $0$, we call it the zero of the semiring $(S, +, \cdot)$. We call $0$ the multiplicatively absorbing if $0 \cdot s$ is absorbing in $(S, \cdot)$, i.e., $\forall x \in S : 0 \cdot x = x \cdot 0 = 0$.

Definition 3.5: Let $A = (A, \cdot)$ be a commutative semigroup and $S = (S, +)$ be a semiring. We call $(S, A, \cdot)$ a left-quasi semimodule over $S$ or $S$-left-quasi-semimodule if there exists a function $S \times A \rightarrow A$ such that for all $r, s \in S$ and $a, b \in A$, we have:

1. $r(a \cdot b) = ra \cdot rb$
2. $r(sa) = (rs)a$

An $S$-left-quasi-semimodule $(S, A, \cdot)$ is called unital if $S$ has an identity $1$ and $\forall a \in A : 1 \cdot a = a$. Furthermore, $(S, A, \cdot)$ is called zero-preserving if there are zeros $0$ and $0$, respectively of $A$ and $S$ that satisfy $\forall a \in A : 0 \cdot a = 0$.

Definition 3.6: Let $(S, A, \cdot)$ be a left-quasi-semimodule. We call $(S, A, \cdot)$ a left-semimodule over $S$ or $S$-left-semimodule if $(r + s)a = ra + sa$.

For simplicity, we use the term quasi-semimodule to denote left-quasi-semimodule and the term semimodule to denote left-semimodule.

Definition 3.7: Let $A$, $B$, and $C$ be sets, and $P$ and $Q$ be relations such that $P \subseteq A \times B$ and $Q \subseteq B \times C$.

- $P \subseteq \{(x, z) | \exists y (y \in B : (x, y) \in P \land (y, z) \in Q)\}$
- $P^\circ \subseteq \{(x, y) | (y, x) \in P\}$

In Definition 3.7, $P \subseteq \{(x, z) | \exists y (y \in B : (x, y) \in P \land (y, z) \in Q)\}$
the properties that involve ciphers and keys based on the operators of the cipher structure.

**Definition 3.11 (Multiplicative Envelope [1]):** Let \( \mathcal{E} \) be a quasi-semimodule over a key structure \( \mathcal{K} \) with an associative operator \( \cdot \) and an identity of \( \mathcal{K} \) (i.e., \( (\mathcal{K}, \cdot, e, 0) \) is a group). We call \( \mathcal{E} \) a multiplicative envelope.

**Definition 3.12 (Additive Envelope [1]):** Let \( \mathcal{E} \) be a zero-preserving semimodule over a key structure \( \mathcal{K} \) where \( + \) is idempotent and \( 0 \) is its identity. We call \( \mathcal{E} \) an additive envelope.

**Definition 3.13 (Envelope [1]):** Let \( \mathcal{E} \) be a system where \( * \) distributes over \( + \) on both the left and right. We call \( \mathcal{E} \) an envelope structure.

**Definition 3.14 (Message [1]):** Let \( \mathcal{M} = (\mathcal{E}, S, \cdot) \) be a unitary zero-preserving semimodule over an envelope structure \( \mathcal{E} \) with an associative operator \( * \), an idempotent operator \( + \), and where \( * \) distributes over \( + \) on both the left and right. We call \( \mathcal{M} \) a message structure.

The message structure represents the encryption and decryption of secrets. The message structure consists of a secret and an envelope. The binary operator in the semimodule represents using an envelope to encrypt or decrypt a secret. The axioms of the envelope and message structures are presented in the appendix.

IV. ANALYSIS OF OBJECT-BASED SCHEMES

Object-based schemes define policies that describe a set of secrets and state the required keys to reveal each one of them. However, these policies should satisfy a predefined properties. For example, assume that we have three secrets \( s_1, s_2, \) and \( s_3 \). Also, assume that we assign three keys to the users of the system as follow: the key \( k_3 \) to Alice, \( k_2 \) to Bob and \( k_3 \) to Carol. Assume that we have the policies:

- Carol should be able to get the secret \( s_1 \). This policy can be specified as
  \[ m_1 := (k_3 \circ a) \cdot s_1. \]
- The secret \( s_2 \) is decrypted twice using first the key of Bob and then the key of Alice. This policy states an order of decryption to get the secret. The encrypted message should be decrypted first by Alice then by the Bob. Here, Bob can get the secret but after the approval of Alice. \( m_2 := (k_1 \circ a) \cdot ((k_2 \circ a) \cdot s_2) \)
- The secret \( s_3 \) can be revealed by using keys of Alice and Bob together, or by using Carol key. \( m_3 := ((k_1 \ast k_2) \cdot k_3) \circ a \cdot s_3 \)

We use our framework with the aid of Prover9 [31] to prove the following properties:

1. The secret \( s_1 \) encrypted using Bob key cannot be produced from the message \( m_3, \) i.e., \( \exists x \in E : (k_2 \circ a) \cdot s_3 \leq x \cdot m_3 \)

Note that the third property can be generalized to any key and any secret from any message. Such generalization would increase the complexity of the analysis. However, our goal in this paper is presenting attacks algebraically on object-based schemes which can be achieved through the presented properties.

We can use our framework to verify the satisfaction of properties. The first property is a correctness property. We were able to verify this property by assuming that \( k_3 \neq 0 \), since the \( 0 \) could not reveal any secret. Also, we were able to prove the second property by having \( k_1 \neq 0 \) and \( k_2 \neq 0 \). Also, to enforce the order of decryption to obtain the secret, we should assume that \( k_1 \neq k_2 \). The third property states that we cannot obtain \( s_3 \) encrypted using \( k_2 \) from the message \( m_3 \). However, the message \( m_3 \) consists of \( s_3 \) encrypted using \( k_2 \) and consists of \( s_3 \) encrypted using \( k_1 \ast k_2 \). Therefore, to prove the third property, we should assume that \( k_2 \neq k_3 \) and combining the keys \( k_1 \) and \( k_2 \) does not produce \( k_2 \) i.e., \( k_1 \ast k_2 \neq k_2 \).

V. ALGEBRAIC MODELS

In this section, we introduce four concrete algebraic models for the algebraic structures presented in Section III-B. These models act as an intermediate level between the abstract level and the implementation level as shown in Figure 1 similar to [12]. Also, these models may include additional algebraic properties that affect the analysis of object-based schemes. We introduce algebraic models for Vigenère, transposition cipher, DES and RSA cryptosystems by giving concrete meaning to secrets, keys, and ciphers. In all these cryptosystems, the envelope operator is represented as applying a key to a cipher, while the operator in the message structure is represented as using an envelope to encrypt or decrypt a secret.

![Different Cryptosystems Model for Object-Based Scheme](image-url)

Figure 1. A diagram that shows the relation between the abstract and concrete models.
A. Vigenère

Vigenère is a substitution cipher such that each alphabet in the plain-text is substituted with another alphabet based on a key. There are two different ways to represent Vigenère algebraically. One of them as a relation and the other as a function. Here we show the functional representation since it is more common. First, we give an example and then we introduce an algebraic model for Vigenère. A secret is a string. For example it can be the word security. The key is also a string e.g., the word test. Each character is encrypted individually by using the function 
\[ c_i = f^{-1}(f(s_i) + f(k_i) \pmod{26}) \]
where \( f(n) \) gives the numerical value of the character \( n \). For example, \( f(a) = 0, f(b) = 1 \), and so on. Note that the key should be repeated if it is shorter than the secret. The produced cipher-text from the given secret and key is “liunkmnlh”. A generalization to this representation can consider each of a secret and a key as a set of strings.

**Proposition 5.1:** Let \( S = (S, +, \cdot, 0) \), where \( S \triangleq \mathcal{P}(A) \) is a set of strings. The operator \( P +, Q \triangleq \mathcal{P} \cup Q \) is a set union. The operator \( P * , Q \triangleq \{ pq \mid p \in P, q \in Q \} \) is the concatenation of strings, \( 0, \triangleq \emptyset \) is the empty set. The structure \( S \) is a secret structure.

**Proof:** The union \( \cup \) is commutative, associative, and idempotent. Its identity element is \( \emptyset \). String concatenation is associative. The operator \( \cdot \) distributes over \( + \), because of the distributivity of \( \cdot \) over \( \cup \). Finally, \( \emptyset \) is an annihilator for \( \cdot \) as \( x \in \emptyset \Leftrightarrow \text{false} \) and \( \{ x \mid \text{false} \} = \emptyset \).

**Proposition 5.2:** Let \( K = (K, +, \cdot, 0) \), where \( K \triangleq \mathcal{P}(A) \) is a set of strings. The operator \( P +, Q \triangleq \mathcal{P} \cup Q \) is a set union. The operator \( P * , Q \triangleq \{ f^{-1}(f(p)) + f(q) \pmod{26} \mid p \in P, q \in Q \} \) is the addition of two strings, \( 0, \triangleq \emptyset \) is the empty set. The structure \( K \) is a key structure.

**Proof:** The union \( \cup \) is commutative, associative, and idempotent. Its identity element is \( \emptyset \). Number addition is commutative and associative. The operator \( * \) distributes over \( + \), because of the distributivity of \( \cdot \) over \( \cup \). Finally, \( \emptyset \) is an annihilator for \( * \) as \( x \in \emptyset \Leftrightarrow \text{false} \) and \( \{ x \mid \text{false} \} = \emptyset \).

The key is also defined as a set of strings. We define the \( +_e \) operator as the set union while the operator \( \cdot_e \) addition of the alphabet values. For example \( \{\text{guke}\} \cdot_e \{\text{nijp}\} = \{\text{test}\} \) i.e., \( f(g) = 6, f(n) = 13 \), and \( f^{-1}(19) = t \) and so on for the other characters.

**Proposition 5.3:** Let \( C \triangleq (C, +, \cdot, 1, \circ, 0) \), where \( C \) is a set of Vigenère ciphers i.e., \( c(s) = s + k \). Let \( c^e \) is a function composition, the operator \( +_e \) be set union, the operator \( \cdot_e \) be identity function, and the operator \( 0_0 \) be the empty set. The structure \( C \) is a cipher structure.

**Proof:** The union \( \cup \) is commutative, associative, and idempotent. Its identity element is \( \emptyset \). \( c(c^e(s)) = s + k = k \). The composition of functions is associative and its identity element is the identity function. The operator \( \cdot_e \), distributes over \( +_e \) because of the distributivity of \( \cdot \) over \( \cup \). Finally, \( \emptyset \) is an annihilator for \( \cdot_e \) as \( x \in \emptyset \Leftrightarrow \text{false} \) and \( \{ x \mid \text{false} \} = \emptyset \).

A cipher is a function that encrypts or decrypts a unit of a text. The identity cipher is the identity function \( c(s) = s \). The \( +_e \) is the set union, and \( 0_0 \) is the empty set. It has been proved in the literature that Vigenère has the following algebraic properties:

1. \( c_1 \cdot_e c_2 = c_2 \cdot_e c_1 \)
2. \( (k_1 \circ_c) \cdot_e (k_2 \circ_c) = (k_1 \cdot_e k_2) \circ_c \)
3. \( e^n = 1_e \)
4. \( 1_e \cdot_e k = k \cdot_e 1_e = k \)

where \( c_1, c_2 \) are envelopes, \( k_1, k_2 \) are keys, and \( c \) is a cipher. We have:

The first identity states that Vigenère is commutative. Identity 2 states that double encryption in Vigenère is equivalent to a single encryption with a combined key. Identity 3 states that Vigenère is cyclic i.e., encrypting a message 26 times (the number of alphabets) using the same keyword produces the same secret. Identity 4 states the existence of an identity key \( 1_e \), i.e., a string of a’s since \( f(a) = 0 \).

Vigenère satisfies the properties of algebraic structures presented in Section III-B. However, as shown above, Vigenère has additional properties. These properties affect the analysis of object-based schemes. When we add these properties to the analysis of the illustrative example, we found that the properties 2 and 3 cannot be verified. Therefore, the scheme contains flaws as shown in the following scenarios.

**Scenario of an Attack:** The second property in the illustrative example states that Alice should decrypt the message first and then Bob in order to get the secret \( s_2 \). Bob can get the secret but it needs the approval of Alice. By having the commutative property of ciphers, we cannot state an order on the decryption process. For example, decrypting using Bob key and then Alice we cannot state an order on the decryption process.
reveals the secret $s_3$ to each one of them. An intruder playing the role of Bob can trick Alice by sending the identity key $1_{\mathbb{K}}$ instead of his key. In this case, the trusted server sends the secret $s_3$, encrypted using the key of Bob, to Alice and Bob. Therefore, Alice cannot get the secret as shown in the derivation below.

$$((k_1 \ast_{\mathbb{K}} k_3) \circ a^* \ast_{\mathbb{K}} ((k_1 \ast_{\mathbb{K}} k_2) \circ a) \ast_{\mathbb{K}} s_3)$$

$$= \langle A \text{ property of an identity key } \rangle$$

$$= \langle \text{Definition 3.12} \rangle$$

$$= \langle \text{Identity of Vigenère } \rangle$$

$$= \langle \text{Definition 3.14} \rangle$$

$$= \langle \text{Definition 3.10} \rangle$$

$$= \langle \text{Definition 3.11} \rangle$$

$$= \langle \text{Definition 3.10} \rangle$$

$$= \langle \text{Definition 3.11} \rangle$$

$$= \langle \text{Definition 3.10} \rangle$$

$$= \langle \text{Definition 3.11} \rangle$$

Therefore, the produced cipher-text is "reseyot". The decryption function is $s_k^{-1}$, where $k^{-1}$ is the inverse of the relation $k$. Our algebraic structures enable us to represent a generalisation to the transposition ciphers. It allows handling a set of secrets and a set of keys as shown in the following propositions.

**Proposition 5.4:** Let $S \triangleq \mathcal{P}(\mathbb{A} \times \mathbb{N})$, where $\mathbb{A}$ is a set of alphabet and $\mathbb{N}$ is the set of natural numbers. Let the operator $P \ast Q \triangleq P \cup Q$ be a set union, the operator $P \ast \ast Q \triangleq \{ p \cap q \mid p \in P \land q \in Q \}$ be the concatenation of strings, and $0 \triangleq \emptyset$ be the empty set. The structure $S = (S, + \ast, \ast, 0)$ is a secret structure.

**Proof:** The proof is similar to the proof of Proposition 5.1.

In this model, a secret is a set of strings each is represented as a relation. The operator $+$ indicates set union, while the operator $\ast$ indicates the concatenation of strings represented as relations.

**Proposition 5.5:** Let $K \triangleq \mathcal{P}(\mathbb{N} \times \mathbb{N})$ where $\mathbb{N}$ is the set of natural numbers. Let the operator $P \ast Q \triangleq P \cup Q$ be a set union, the operator $P \ast \ast Q \triangleq \{ p \cap q \mid p \in P \land q \in Q \}$ be the intersection of relations, and $0 \triangleq \emptyset$ be the empty set. The structure $K = (K, + \ast, \ast, 0)$ is a key structure.

**Proof:** The union $\cup$ is commutative, associative, and idempotent. Its identity element is $\emptyset$. Set intersection $\cap$ is commutative and associative. The operator $\ast$ distributes over $+$ because of the distributivity of $\land$ over $\lor$. Finally, $\emptyset$ is an annihilator for $\ast$ as $x \in \emptyset$ is false and $\{ x \mid x \}$ is false.

Our key model allows representing a set of keys and handling two operators. The operator $+$ is the set union and the operator $\ast$ is the set intersection. Note that this is one representation of keys. There could be other representations that satisfy the properties of the key structure.

**Proposition 5.6:** Let $C \overset{\triangleq}{=} (C, + \ast, +, 1, 0)$, where $C$ is a set of transposition ciphers i.e., $c(s) = s \ast k$. Let $c^j(s) = s \ast k^j$, $P \ast Q \triangleq \{ p \cap q \mid p \in P \land q \in Q \}$ where $\ast$ is a relation composition, the operator $+$ be set union, the operator $1$ be the identity relation, and the operator $0$ be the empty set. The structure $C$ is a cipher structure.

**Proof:** The union $\cup$ is commutative, associative, and idempotent. Its identity element is $\emptyset$. $c, c^1 \ast c^j = s \ast k; k^j \ast c = s \ast k^j; k = s$. The composition of relations is associative and its identity element is the identity relation. The operator $\ast$ distributes over $+$, because of the distributivity of $\land$ over $\lor$. Finally, $\emptyset$ is an annihilator for $\ast$ as $x \in \emptyset$ is false and $\{ x \mid x \}$ is false.

Transposition ciphers do not have extra algebraic properties that could affect the security properties of object-based scheme presented in the example. However, we do
not claim in this paper that transposition cipher is secure. It is known that transposition ciphers are vulnerable to attacks based on frequency analysis. Our focus of this paper is analysing the effects of the algebraic properties of cryptosystems on object-based schemes. We do not discuss the security of cryptosystems themselves.

C. Data Encryption Standard (DES)

DES is a shared key cryptosystem such that the key used in encryption is the same used in decryption. DES is a block cipher that takes a block of plain-text as input and produces a cipher-text of the same length. DES is based on sixteen rounds. Each round consists of four stages that perform expansion, key mixing, permutation, and substitution. In our representation, we denote the encryption algorithm by $c$ and the decryption algorithm by $c'$. We represent secrets and keys as sets of numbers.

Proposition 5.7: Let $S \triangleq \mathcal{P}(\mathcal{N})$, where $\mathcal{N}$ is the set of natural numbers. Let the operator $P \cup Q \triangleq P \cup Q$ be a set union, the operator $P + Q \triangleq \{p \oplus q \mid p \in P \land q \in Q\}$ be the concatenation of strings represented as numbers, and $0 \triangleq \emptyset$ be the empty set. The structure $S = (S, +, *, 0)$ is a secret structure.

Proof: The proof is similar to the proof of Proposition 5.1.

Proposition 5.8: Let $K \triangleq \mathcal{P}(\mathcal{N})$, where $\mathcal{N}$ is the set of natural numbers. Let the operator $P + Q \triangleq P \cup Q$ be a set union, the operator $P \cdot Q \triangleq \{pq \mid p \in P \land q \in Q\}$ be the multiplication of keys, and $0 \triangleq \emptyset$ be the empty set. The structure $K = (K, +, *, 0)$ is a key structure.

Proof: The union $\cup$ is commutative, associative, and idempotent. Its identity element is $\emptyset$. The xor operator $\oplus$ is commutative and associative. The operator $*$ distributes over $\oplus$ because of the distributivity of $\land$ over $\lor$. Finally, $\emptyset$ is an annihilator for $*$, as $x \cup \emptyset \equiv false$ and $\{x \mid false\} = \emptyset$.

Proposition 5.9: Let $C$ be a set of DES algorithms, the operator $*$ represents running algorithms consecutively, the operator $+$ be set union, the operator $\cdot$ be an algorithm that has not effect, and the operator $0$ be the empty set.

Proof: Set union $\cup$ is commutative, associative and idempotent with an identity $\emptyset$. The operator $*$ is associative with an identity $1$.

Scenario of an Attack: The presented key model has the following property

$$k \cdot s = 1$$

This property leads to the following derivation.

$$((k_1 \cdot s) \cdot k_1) \cdot a * a) \cdot s_3$$

$$= \{A property of the key model\}$$

$$= \{1_{k_1} \cdot s_3\}$$

$$= \{A property of the key model\}$$

$$= \{1_{k_1} \cdot s_3\}$$

$$= \{\text{Definition 3.11}\}$$

$$1_{k_1} \cdot s_3 + _{s_3} \cdot (k_3 \cdot a) \cdot s_3$$

$$= \{\text{Definition 3.12}\}$$

$$= \{\text{Definition 3.14}\}$$

By having the $k_1 = k_2$, we can reveal the secret which violates the secrecy property. Therefore, we should assume that $k_1 \neq k_2$.

Scenario of an Attack: DES has the following property, when the keys $k_1$ and $k_2$ are semi-weak keys.

$$(k_1 + a) \cdot ((k_2 \cdot a) \cdot s) = s$$

This property affects the satisfaction of the second property of the illustrative example as shown in the following derivation.

$$(k_3 + a) \cdot ((k_2 \cdot a) \cdot s)$$

$$= \{\text{A property of semi-weak keys}\}$$

$$s$$

Therefore, the secret $s_2$ would be revealed without decrypting the message using the keys of Alice and Bob. Therefore, to prove the property, we should make sure that the used keys are not semi-weak.

D. RSA

RSA is a public key cryptosystem whose key consists of two parts: a public part $(e, n)$ used for encryption, and a private part $(d, n)$ used for decryption. The encryption of a secret $s$ is performed as $s^e \mod n$ while the decryption of a message $m$ as $m^d \mod n$.

Proposition 5.10: Let $S \triangleq \mathcal{P}(\mathcal{N})$, where $\mathcal{N}$ is the set of natural numbers. Let the operator $P + Q \triangleq P \cup Q$ be a set union, the operator $P \cdot Q \triangleq \{pq \mid p \in P \land q \in Q\}$ be the concatenation of strings represented as numbers, and $0 \triangleq \emptyset$ be the empty set. The structure $S = (S, +, *, 0)$ is a secret structure.

Proof: The proof is similar to the proof of Proposition 5.1.

In this model, a secret is a set of numbers. The operator $+$ indicates set union while the operator $*$ indicates the concatenation of strings.

Proposition 5.11: Let $K \triangleq \mathcal{P}(\{(a, b, m) \mid a, b \in \mathbb{N}, m \in M \land 1 < a < \phi(m) \land \gcd(a, \phi(m)) = 1 \land a \times b \equiv 1 \mod \phi(m)\})$ where $\mathcal{N}$ is the set of natural numbers. Let the operator $P + Q \triangleq P \cup Q$ be a set union, the operator $P \cdot Q \triangleq \{pq \mid p \in P \land q \in Q\}$ be the multiplication of keys, and $0 \triangleq \emptyset$ be the empty set. The structure $K = (K, +, *, 0)$ is a key structure.

Proof: The union $\cup$ is commutative, associative, and idempotent. Its identity element is $\emptyset$. Number multiplication is commutative and associative. The operator $*$,
distributes over $\lor$, because of the distributivity of $\land$ over $\lor$. Finally, $\emptyset$ is an annihilator for $\ast$, as $x \in \emptyset \iff \text{false}$ and $\{x \mid \text{false} \} = \emptyset$.

An RSA key consists of three parts: $e$, $d$, $n$. In our structure, a key is a set of RSA keys. The operator $\lor$ is a set union, while the operator $\ast$ represents combining keys that have the same value $n$ by multiplying together the values of $d$ and the values of $e$.

**Proposition 5.12:** Let $C$ be a set of RSA ciphers, the operator $\ast$ be a function composition, the operator $\lor$ be set union, the operator $1$ be the identity function, and the operator $0$ be the empty set.

**Proof:** The proof uses properties of set and function theories.

It has been proved in the literature that RSA has the homomorphism algebraic property

$$e \cdot s_1 \ast e \cdot s_1 = e \cdot (s_1 \ast s_2)$$

By analysing the object-based scheme presented in the illustrative example, we find that this property does not affect its security property.

**VI. Conclusion**

In this paper, we extend the work of [1] in analysing object-based schemes algebraically in two directions. First, we present concrete algebraic models for Vigenère, transposition ciphers, DES and RSA cryptosystems. These models act as an intermediate level between the abstract level and the implementation. Second, we show the effects of the additional algebraic properties of those concrete models on the security properties of object-based schemes. We find that some algebraic properties of cryptosystems such as commutativity could affect the security of object-based schemes based on the scheme and the security properties.

As a future work, we aim at building a tool that can be used for automatically verify security policies. The input to the tool are security properties and an object based scheme. The security properties can be represented as logic formulas, while the object based schemes are set of policies that can be represented as terms of the algebraic structures presented in this paper. The tool should store the algebraic properties of several cryptosystems. By using prover9, the tool should be able to indicate automatically which property is satisfied and which one is not satisfied. Such a tool would enhance the security of systems implementing access control based on object-based schemes.

**APPENDIX**

**A. Axioms of the Envelope Structure**

1. $\forall (a, k, l \mid a \in C, k, l \in K : (k + l) a = k a + l a)\n2. \forall (a, b, k \mid a, b \in C, k \in K : k (a + b) = k a + k b)\n3. \forall (a, b, k \mid a, b \in C, k \in K : k (a \ast b) = k a \ast k b)\n4. \forall (a, k, l \mid a \in C, k, l \in K : l (k a) = (l \ast k) a)\n5. \forall (a \mid a \in C : 0 a = 0)\n6. \forall (k \mid k \in K : k 0 = 0)\n7. \forall (k \mid k \in K - \{0\} : k 1 = 1)\n8. \forall (k, a \mid k \in K, a \in A : k a' = (k a)')\n9. \forall (a, k, l \mid a \in C, k, l \in K : (k + l) a = (l + k) a)\n10. \forall (a, k, l, h \mid a, k, l, h \in K : ((k + h) l) a = ((l + h) k) a)\n11. \forall (a, k \mid a \in C, k \in K : (k + k) a = k a)\n12. \forall (a, k, l \mid a \in C, k, l \in K : (k + l) a = (l + k) a)\n13. \forall (a, k, l, h \mid a, k, l, h \in K : ((k + l) \ast k) h a = ((k \ast l) h) a)\n14. \forall (a, k, l \mid a \in C, k \in K : (k + k) a = 0 a)\n15. \forall (a, k, l \mid a \in C, k, l, h \in K : ((k + l) \ast k) h a = (k \ast l) (l + k) a)\n16. \forall (a, k, l, h, a \mid k, l, h, k a \in C : (k + h) a = (k \ast l) (l + h) a)\n17. \forall (a, b, k \mid a, b, c, k \in K : k (a + b) = k (a \ast (b + c)))\n18. \forall (a, b, k \mid a, b, c, k \in K : k (a \ast b) \ast c = k (a \ast (b + c)))\n19. \forall (a, k, l \mid a \in C, k \in K : k (a + k) a = k a)\n20. \forall (a, k, l \mid a \in C, k \in K : k (a + k) 0 a = k a)\n21. \forall (a, k \mid a \in C, k \in K : k (a + k) 0 a = k a)\n22. \forall (a, k \mid a \in C, k \in K : k (a + k) 0 a = k a)\n23. \forall (a, k \mid a \in C, k \in K : k (a + k) 0 a = k a)\n24. \forall (a, k \mid a \in C, k \in K : k (a + k) 0 a = k a)\n25. \forall (a, k \mid a \in C, k \in K : k (a + k) 0 a = k a)\n26. \forall (a, k \mid a \in C, k \in K : k (a + k) 0 a = k a)\n27. \forall (a, k, l, h \mid a, b, c, k \in C : k (a + b) \ast c = k ((a + b) \ast c))\n28. \forall (a, b, c \mid a, b, c \in C : k (a + b) \ast c = k ((a + b) \ast c))$
18) $\forall (a, s, r, t \mid a \in C, s, r, t \in S : a((s + r) + t) = a(s + (r + t)))$

19) $\forall (a, s \mid a \in C, s \in S : a(s + 0) = as)$

20) $\forall (a, s \mid a \in C, s \in S : a(s + s) = as)$

21) $\forall (a, s, r, t \mid a \in C, s, r, t \in S : a((s + r) + t) = a(s + (r + t)))$

22) $\forall (a, s, r, t \mid a \in C, s, r, t \in S : a(s + (r + t)) = a(s + r) + (t + s))$

23) $\forall (a, s, r, t \mid a \in C, s, r, t \in S : a((s + r) + t) = a(s + (r + t)))$

REFERENCES


Khair Eddin Sabri has been working as an assistant professor in the Computer Science Department at the University of Jordan since 2010. He obtained his B.Sc. degree in Computer Science from the Applied Science University, Jordan in June 2001. He also received M.Sc. degree in Computer Science from the University of Jordan in January 2004 and a Ph.D. degree in Software Engineering from McMaster University, Ontario Canada in June 2010. He is a member of the Formal Requirements and Information Security Enhancement (FRAISE) Research Group. His main research interest is the formal verification and analysis of security properties.