WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Innovative Enhancement of the Playfair Cipher for Cryptography

PRAJAPATI PRITESH ASHVINBHAI[1], PROF. VIJAYSINH JADEJA[2], RIBADIYA DHRUVIL RASIKBHAI[3], PANCHAL VEDANT ROHITBHAI[4],MEHTA ROHAN SUNILBHAIL[5],RANA VEERPALSINH BAHADURSINH[6]

[1] SAL COLLEGE OF ENGINEERING,
[2] DEPARTMENT OF ENGINEERING,
[3] AHMEDABAD, GUJARAT, INDIA.

ABSTRACT :

The paper gives an idea about a highly modified version of Classical Playfair cipher with an aim for the enhanced features like higher confidentiality, remaining resistant with the known classical attacks, and at the same time, being computationally lightweight for resource-constrained environments. The enhancement made Three major changes: (1) Ephemeral matrix seed exchange inspired by Diffie-Hellman to obtain per-session base key and, (2) The matrix evolution off a Playfair 5×5 key matrix that updates after every block (digraph) encryption the reversible matrix transformations were used, and (3) A position-dependent masking of digraph coordinates using a keystream derived method. If combined these modifications will lead to the significant cipher effective keyspace, open way for the breaking of periodic statistical patterns identified by digraph frequency analysis, and in addition, deliver forward secrecy with the use of ephemeral seeds. The full algorithm explanation, examples, security and complexity analysis, and also recommendations related to parameter selection are part of the content of this paper.

Keywords: Playfair cipher, symmetric cryptography, key evolution, ephemeral key exchange, digraph masking, forward secrecy

Introduction

Playfair cipher, which was developed in the 19th century is a type of digraph substituion cipher in which pair letters are enciphered using a key matrix (or key table). While long obsolete by contemporary high-security needs, Playfair is educational and potentially useful for low-resource situations due to its simplicity and rapidity. However, classic Playfair is susceptible to digraph frequency analysis and known-plaintext attacks as its key matrix is static throughout the whole message session.

In this paper, we present a new enhancement to Playfair that maintains its performance while increasing its resistance to classical ciphertext attacks. The approach is constructed on three complimentary principles:

1.Memorable Matrix Seed Exchange: memorandum-inspired full-key exchange with simple seeds (like Diffie–Hellman but with matrix seeds) to generate per-session keys on every secret that does not depend on large primes and supports forward secrecy running easily when secrets are ephemeral.
2.Over each block, the matrix evolves—the plaintexts are augmented one at a time by the reversible linear and permutation operations applied to an original set of blocks in order; and each transformation adds another layer of security because once we encrypt any digraph, we have modified the key matrix deterministically, so that if you treat this ciphertext as produced by a single static substitution table, one would not be able to decipher it next time.

3. Mask coordinates with keystream — the row/column coordinates of letters are masked using a short pseudorandom keystream computed from the transient seed and past ciphertext, disrupting digraph frequency correlations.

We demonstrate how to implement these components securely, give a full algorithm, and address security, complexity, and usage advice.

Context

A. The Traditional Playfair Cryptosystem

A keyword is used by Playfair to create a 5x5 matrix (I/J combined), and the remaining letters are filled in alphabetically. These guidelines are applied when processing each plaintext digraph:

Replace each letter with the letter to its right (wrap around) if they are in the same row.

Replace each with the letter beneath it (wrap around) if they are in the same column.

If not, use the rectangle rule to swap out each letter for the one in the same row but in the other letter's column.

The plan is completed with padding and repeated letter handling. Static mapping and digraph frequency leakage are the main causes of Classic Playfair's shortcomings.

B. Encouragement to Improve

In order to make Playfair better, we want to: (1) evolve the key matrix to prevent static matrix attacks; (2) use a keystream to mask predictable coordinate patterns; and (3) optionally enable session forward secrecy using ephemeral seeds. The improvements should only require a small amount of additional memory and maintain computation O(n) in plaintext length.

Design Objectives and the Threat Model

A passive eavesdropper with the ability to record ciphertext and potentially recognized plaintext/ciphertext pairs is the threat model. When used as black boxes, we assume that the adversary cannot crack secure hash primitives or standard modular arithmetic.
Design goals:

Expanded effective keyspace: Use useful resources to render exhaustive key recovery impossible.

Break stationary digraph distributions to defend against statistical digraph attacks.

Minimal overhead: Preserve low memory usage and linear encryption time.

Optional forward secrecy: If ephemeral seeds are used, previous session matrices shouldn't be exposed in the event that a long-term secret is compromised.

Proposed Algorithm: Enhanced Playfair (E-Playfair)

A. High-level Summary

1. Using a lightweight authenticated exchange (which can be Diffie-Hellman-like or a pre-shared long-term secret plus a nonce), two parties generate a shared 32-bit or 64-bit ephemeral seed. A pseudorandom function (PRF) and base key material are initialized by the seed S.

2. Create an initial 5×5 Playfair matrix M0 (I/J merged) from S and a user keyword K using a deterministic but salted ordering obtained from PRF(S || K).

3. To apply the Playfair substitution on the current matrix M_i, first obtain a short keystream value ks_i = PRF(S || counter || last_cipher_block) truncated to a small integer for each plaintext digraph P_i (letters a,b). Then, use ks_i to mask the row/column coordinates of the pair.

4. After creating cipher digraph C_i, apply a reversible transform T parameterized by S and i to update the matrix M_{i+1} (e.g., rotate rows by a derived offset, perform a small permutation of rows/columns, or apply a modular reindexing). Although the transform is selected to allow for deterministic computation by both parties, an attacker may perceive it as pseudorandom.

B. Primitives and notation

PRF(x) is a secure pseudorandom function (truncated versions of AES-CMAC or HMAC-SHA256). A lightweight KDF/stream, such as HKDF with SHA-256, is appropriate for devices with limited resources.

Ephemeral seed S (64 bits or more is recommended). S = DH_shared_value when true Diffie–Hellman is applied.

K is the user keyword string (optional, for human memorability). Matrix ordering is obtained by combining with S.

The i-th digraph is represented by the Playfair 5x5 matrix M_i.

counter: an integer digraph index with a starting value of 0.

For coordinate masking, ks_i is a small integer obtained from PRF (range 0..4 for rows/cols or 0..24 for linear shift, depending on method).

The reversible matrix update function is T(S, i, M).

C. Matrix construction (M0)

1. Determine seed_bytes = PRF(S || 0x01 || K).

2. Use seed_bytes to specify an order for the 25 letters (A–Z, excluding J). The bytes are interpreted as an 8-bit number stream, and a Fisher–Yates style mapping seeded from the stream is used to select letters without replacement. That is, to create M0, start with the alphabetical list L, which consists of 25 letters, and shuffle it deterministically using seed_bytes.

Because S alters the matrix ordering, this prevents simple keyword guessability.

D. Coordinate masking and digraph encryption

For every digraph index i:

1. Let P_i = (p1, p2) be the plaintext digraph following the standard Playfair preprocessing (replace J→I, insert filler for repeated letters, pad the final digraph if necessary).
2. Determine ks_i = (PRF(S || 0x02 || i || last_C) mod 5), where last_C is the previous ciphertext digraph encoded as a small integer (or 0 for i=0). ks_i is zero.4.

3. Locate p1 and p2's coordinates (r1,c1) and (r2,c2) in matrix M_i.

4. Mask coordinates are as follows: r1' = (r1 + ks_i) mod 5, c1' = (c1 + ks_i) mod 5, r2' = (r2 + ks_i) mod 5, and c2' = (c2 + ks_i) mod 5.

5. Create masked ciphertext letters c1_mask, c2_mask by applying traditional Playfair substitution with masked coordinates on M_i.

. 6. To ensure that the ciphertext letters are drawn from the original matrix coordinates that have been adjusted back, reverse the additive mask on the generated ciphertext coordinates before outputting. In specifics: determine the coordinates of c1_mask and c2_mask in M_i as (R1,C1), (R2,C2), then calculate the output coordinates (R1 - ks_i mod 5, C1 - ks_i mod 5), map these to letters in M_i, and output.

This masking step ensures that different positions (or sessions) of the same plaintext pair result in different effective coordinate operations, confusing digraph statistics.

E. Matrix evolution (T)

Compute M_{i+1} = T(S, i, M_i) after digraph i has been encrypted. T can be any reversible operation chosen from a small family parameterized by PRF(S || 0x03 || i) so that both parties can compute it deterministically.

From a control byte b (0..255), a practical T can be composed of basic transforms:

Rotate rows downward by k mod 5 using row_rotate(k).

Rotate columns to the right by k mod 5 using col_rotate(k).

swap_rows(a,b): switch rows a and b (a,b is derived from b).

swap_cols(a,b): exchange columns a and b.

Transpose the matrix using transpose().

Utilize the first byte, b = PRF(S || 0x03 || i)[0]. Select and parameterize a brief series of transforms by interpreting bits, then apply them in a predetermined order. The receiver can mirror the same updates during decryption since transforms are deterministic given S and i and reversible.

F. The process of decryption

Encryption and decryption are similar:

1. The parties build M0 in the same way and share S and K.

2. For digraph index i, find the ciphertext letters in M_i, apply the inverse Playfair rules with coordinate unmasking, compute ks_i in the same manner (the decryptor uses last_C from the previously decrypted block), and then update the matrix M_{i+1} by applying T(S, i, M_i).
A deterministic encoding of the previous ciphertext digraph (e.g., linear index 0..24 for pair) must be decided upon; care must be taken to ensure that the computation of ks_i on both sides uses the
same last_C representation.

Illustration

The parameters are, for instance, S = 0xA1B2C3D4 and keyword K = "NETWORK"..

1. PRF(S||0x01||K) and deterministic seeded alphabet shuffle (I/J merge) are used to create M0; the resulting matrix (illustrative) looks like this:

N E T W O
R K A B C
D F G H I
L M P Q S
U V X Y Z

2. Plaintext: "HELLO WORLD" → preprocess to digraphs: HE LX LO OW OR LD (if necessary, apply the J→I rule and use filler X for double L).

3. For example, ks_0 = PRF(...) mod 5 = 2 for i=0. Find H and E, mask the coordinates, apply the Playfair rules to M0, unmask the output, and create the first digraph of the ciphertext.

4. Update to M1 = T(S,0,M0) — for example, swap columns 0 and 3 and rotate rows by 1.

5. Proceed to the next digraph; subsequent digraph encryptions employ a different mapping since M varies.
(For the sake of conciseness, a concrete numerical walkthrough is not included here; however, it can be added upon request.)

Analysis of Security

A. Greater unpredictability and keyspace
The per-session seed S and the matrix evolution path effectively multiply the static Playfair keyspace (25! possible matrices constrained by keyword structure). Now, an attacker has to recover the transformation sequence in addition to the initial matrix. The effective search space becomes unfeasible for brute force methods when the seed is sufficiently large and the PRF is robust.

B. Opposition to digraph frequency analysis
Coordinate masking disrupts digraph distribution stationarity: based on ks_i and the current matrix M_i, the same plaintext digraph maps via distinct masked coordinates. Long-range digraph patterns are further destroyed by matrix evolution. As a result, standard digraph frequency tables lose their usefulness.

C. Attacks using known plaintext and chosen plaintext
Since the matrix changes after each block, matrix evolution limits the amount of information that a limited number of known plaintext/ciphertext pairs can reveal about future mappings to an attacker during a session. Limit session lifetimes (short S lifetime) and rate-limit encryption requests are two ways to mitigate the risk of the attacker reconstructing transformation rules if they are able to request multiple selected plaintexts in a single session (i.e., an adaptive oracle).

D. Confidentiality of forwarding
Compromise of long-term keys does not expose the matrices of previous sessions if S is derived from an ephemeral DH exchange. The per-session S is necessary to reconstruct the actual matrices used, even if the attacker manages to recover the keyword K later.

E. Operational warnings
The PRF and S's confidentiality are essential to the design. To prevent active MITM attacks on seed exchange, use authenticated key agreement.

To prevent biasing ks_i values, PRF output truncation must be done carefully; if PRF is cryptographically strong, modulus 5 can be used.

Complexity and Performance

Time complexity: O(n) for n letters (message length linear) — each digraph needs a fixed number of PRF calls, coordinate lookups in a 5x5 matrix, and a minimal number of swaps/rotations for matrix evolution.

Beyond storing the 5x5 matrix and a small constant state (counters, last ciphertext), the space complexity is O(1). There is very little memory overhead.

PRF evaluation is the primary expense on contemporary hardware or microcontrollers with limited resources. A lightweight stream cipher (such as the Salsa20 core for PRF) or truncated AES-CMAC can be used in place of HMAC-SHA256 if it is too heavy.

Experimental Evaluation

The precise timings vary depending on the implementation language and PRF selected, but this paper presents a proposed experimental methodology.

1. Setup: Use E-Playfair with a lightweight PRF, E-Playfair with HMAC-SHA256, and classic Playfair.

2. Metrics: memory footprint, encryption/decryption throughput (KB/s), and statistical measures of the distribution of ciphertext digraphs (e.g., KL divergence vs. uniform and vs. classic Playfair).

3. Datasets: measure the rate at which digraph frequencies converge using plaintext samples of varying lengths (larger synthetic corpora and small messages typical for Playfair).

Anticipated outcomes: E-Playfair significantly lowers digraph frequency correlations (KL divergence drops) and withstands static matrix recovery attacks for realistic message lengths, despite incurring a slight increase in CPU costs for PRF calls and matrix transforms.

Advantages, Limitations, and Use Cases

Benefits:
include a significant expansion of the effective keyspace and useful defense against traditional attacks.

When using ephemeral seeds, maintain secrecy.

Block ciphers are not desired in embedded systems due to their low memory footprint and linear runtime.

Limitations:
For high-security or high-volume requirements, it cannot be used in place of contemporary symmetric ciphers (AES).

PRF quality and seed exchange confidentiality/authentication are key components of security.
If the session lifetime is long, it could be susceptible to strong adaptive chosen-plaintext attackers.

Suggested applications include educational resources, secure low-bandwidth telemetry, and restricted settings where a balance between enhanced security and ease of use is preferred.

Conclusion

We have introduced E-Playfair, an improved Playfair cipher that combines deterministic reversible matrix evolution, coordinate masking by a PRF-derived keystream, and ephemeral seeded matrix construction. These modifications enable optional forward secrecy, greatly increase resistance to digraph frequency analysis, and maintain Playfair's speed and simplicity. The plan is useful and adaptable: PRF selection and seed length can be changed to accommodate specific implementation limitations. A formalization of security bounds and careful cryptanalysis under adaptive chosen-plaintext models are examples of future work.

REFERENCES

1. Deepthi, R. ”A survey paper on Playfair cipher and itsvariants.” Int. Res. J. Eng. Technol 4.4: 2607-2610.
2. Wang, Yuzhe. ”A Classical Cipher-Playfair Cipher and Its Improved Versions.” International Conference on Electronic Information Engineering and ComputerScience (EIECS).
3. Maha, Maherin Mizan, Md Masuduzzaman, and Ab- hijit Bhowmik. ”An effective modification of play fair cipher with performance analysis using 6X6 matrix.” Proceedings of the International Conference on Com- puting Advancements.
4. Curtin M., “Cracking the Data Encryption”, Springer
5. Sharma, Nikhil, et al. ”A review on playfair substitu- tion cipher and frequency analysis attack on playfair.”
6. Bhowmick, Anirban, Anand Vardhan Lal, and Nitish Ranjan. "Enhanced 6x6 Playfair Cipher using Double Myszkowski Transposition."International Journal of Engineering Research and Technology.