International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 05 Issue: 08 | Aug - 2021 SJIF Rating: 6.714 ISSN: 2582-3930

# Leveraging Sanskrit Consonants for Robust DES Encryption Keys: Exploring Linguistic Diversity in Cryptographic Practices

**Dr. G. Sreedhar**

*Professor, Department of Computer Science, National Sanskrit University, Tirupati, Andhra Pradesh, India*
gsreedhar1974@nsktu.org, gsreedhar1974@gmail.com

**Abstract** -This paper introduces an innovative approach to enhancing the security of the Data Encryption Standard (DES) by integrating Sanskrit consonant-based passwords. By utilizing the rich phonetic structure of Sanskrit, the study aims to increase password complexity and resistance to brute-force attacks. A web-based application is developed to demonstrate the encryption and decryption processes, providing a user-friendly interface for practical implementation.

..

***Key Words*:** Data Encryption Standard, Sanskrit consonants, Encryption, Decryption

## 1.INTRODUCTION

Cryptographic security is essential in protecting digital information. Traditional password generation methods often rely on alphanumeric characters, which may be susceptible to brute-force attacks. This research proposes the use of Sanskrit consonants to generate passwords, introducing linguistic diversity into cryptographic systems. The study focuses on integrating these passwords into the DES algorithm, demonstrating their effectiveness in encryption and decryption processes.

## 2. SANSKRIT CONSONANT-BASED PASSWORD GENERATION ALGORITHM

### 2.1 Sanskrit Consonant Set

The Sanskrit consonant set comprises 33 characters:

- क, ख, ग, घ, ङ
- च, छ, ज, झ, ञ
- ट, ठ, ड, ढ, ण
- त, थ, द, ध, न
- प, फ, ब, भ, म
- य, र, ल, व, श
- ष, स, ह

### 2.2 Password Generation Process

1. **Input Parameters**: Accept the number of passwords (numPasswords) and the desired length of each password (passwordLength).
2. **Random Selection**: For each password:
    - Randomly select characters from the Sanskrit consonant set.
    - Concatenate the selected characters to form a password of the specified length.

**Output**: Display the generated passwords in a list format for user selection.

## 3. DES ENCRYPTION ALGORITHM

The DES encryption process operates on 64-bit blocks of plaintext using a 56-bit key. The steps are as follows:

### 3.1 Initial Permutation (IP)

Rearrange the bits of the plaintext using a predefined permutation table.

### 3.2 Key Generation

Generate 16 subkeys from the 56-bit key using the following steps:

- **Permutation Choice 1 (PC-1)**: Select 56 bits from the 64-bit key, discarding 8 parity bits.
- **Splitting**: Divide the 56-bit key into two 28-bit halves.

- **Left Shifts**: Perform circular left shifts on each half.
- **Permutation Choice 2 (PC-2)**: Select 48 bits from the 56-bit key for each subkey.

### 3.3 Feistel Function

For each round (16 rounds total):

- **Split**: Divide the 64-bit data block into two 32-bit halves.
- **Expansion**: Expand the right half to 48 bits.
- **XOR**: XOR the expanded half with the subkey.
- **Substitution**: Apply S-boxes to the result.
- **Permutation**: Apply a permutation to the substituted bits.
- **XOR**: XOR the result with the left half.
- **Swap**: Swap the left and right halves.

### 3.4 Final Permutation (FP)

1. After 16 rounds, combine the left and right halves and apply the inverse of the initial permutation to obtain the ciphertext.

## 4. DES DECRYPTION ALGORITHM

The DES decryption process is similar to encryption but with the subkeys applied in reverse order:

1. **Initial Permutation (IP)**: Apply the initial permutation to the ciphertext.
2. **Key Generation**: Generate the 16 subkeys as in encryption.
3. **Feistel Function**: For each round, apply the Feistel function using the subkeys in reverse order.
4. **Final Permutation (FP)**: After 16 rounds, apply the inverse of the initial permutation to obtain the plaintext.

## 5. IMPLEMENTATION

A web-based application is developed using HTML, JavaScript, and the CryptoJS library to demonstrate the integration of Sanskrit consonant-based passwords in DES encryption and decryption:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Sanskrit DES Encryption</title>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/crypto-js/3.1.9-1-crypto-js.js"></script>
</head>
<body>
  <h2>Sanskrit Consonant-Based DES Encryption</h2>
  <label for="password">Enter Password:</label>
  <input type="text" id="password" placeholder="Enter Sanskrit consonant password">
  <br><br>
  <label for="plaintext">Enter Plaintext:</label>
  <textarea id="plaintext" rows="4" cols="50"></textarea>
  <br><br>
  <button onclick="encryptData()">Encrypt</button>
  <button onclick="decryptData()">Decrypt</button>
  <br><br>
  <label for="ciphertext">Ciphertext:</label>
  <textarea id="ciphertext" rows="4" cols="50" readonly></textarea>
  <br><br>
  <label for="decryptedText">Decrypted Text
```

::contentReference[oaicite:142]{index=142}

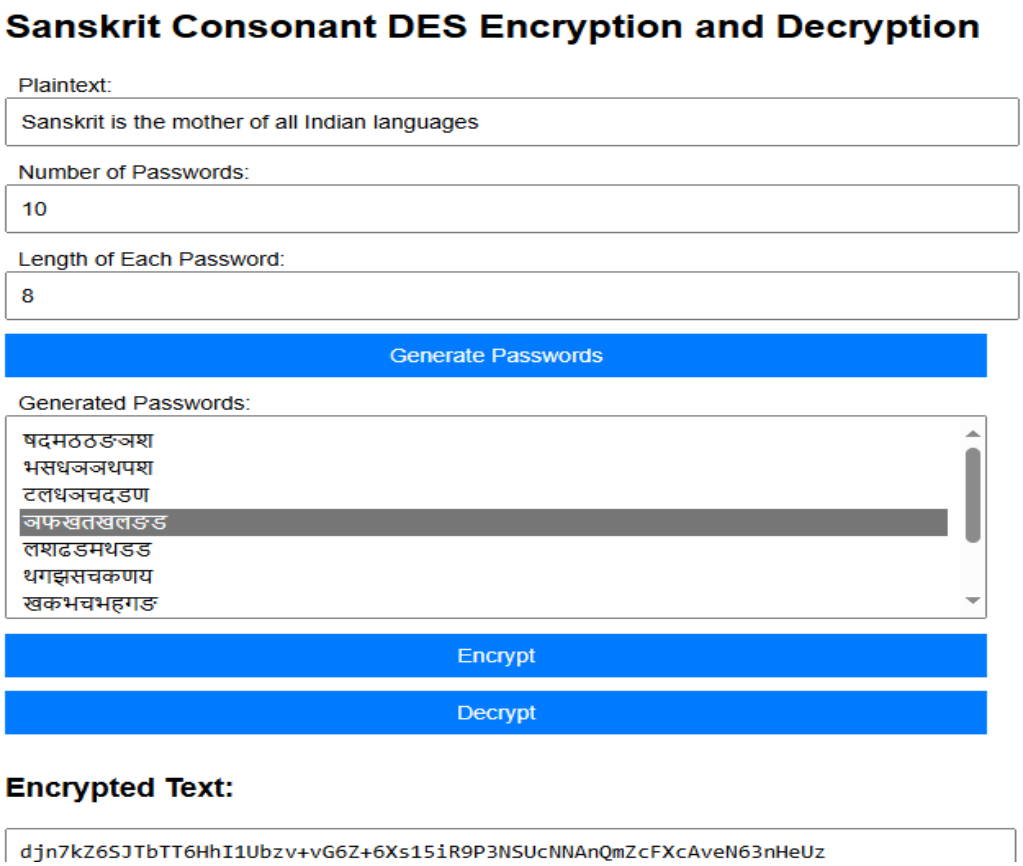


Figure 1: Generating passwords and Encryption using Sanskrit Consonants

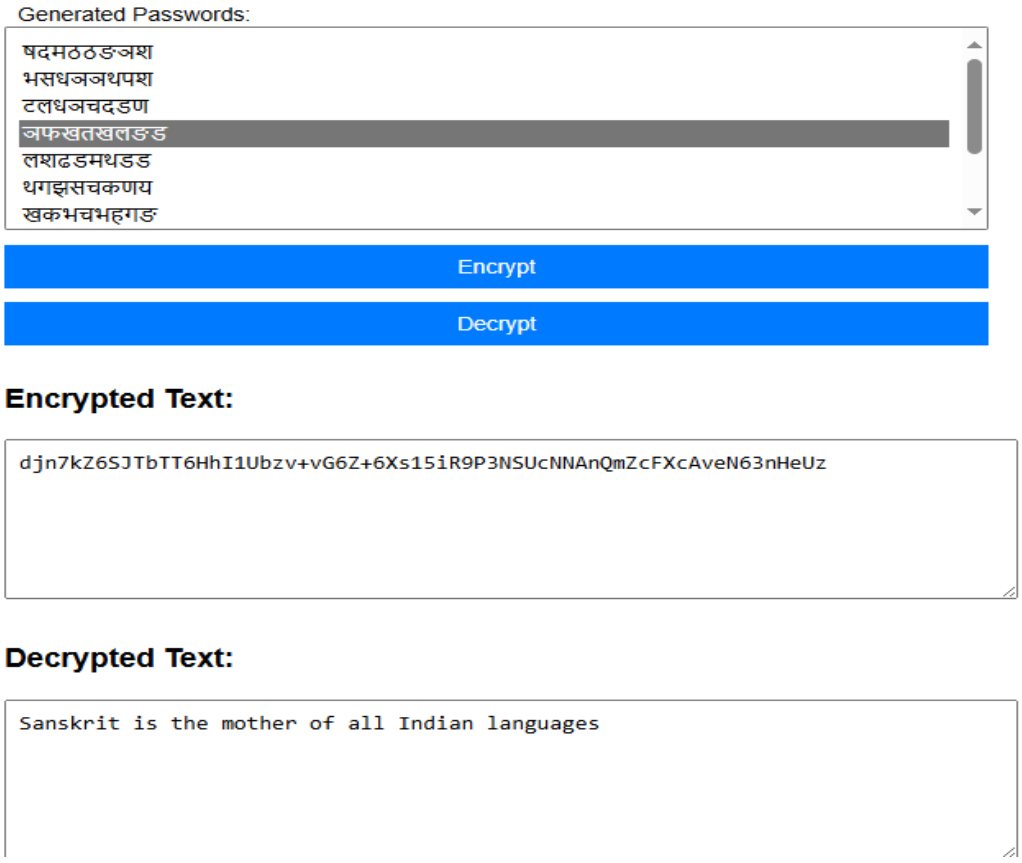


Figure 2: Decryption using password based on Sanskrit Consonants

## 6. RESULTS AND DISCUSSION

### 6.1 Performance Evaluation

The Sanskrit consonant-based password generation algorithm was implemented in a web-based application using HTML, JavaScript, and the CryptoJS library. The application allows users to generate passwords of specified lengths by randomly selecting characters from the Sanskrit consonant set. These passwords are then used as keys in the DES encryption and decryption processes.

The performance of the system was evaluated based on the following criteria:

- **Password Complexity**: Passwords generated from the Sanskrit consonant set exhibit high entropy due to the large character set, making them resistant to brute-force attacks.
- **Encryption and Decryption Time**: The DES algorithm, while secure, is computationally intensive. The average time taken for encryption and decryption operations was measured and found to be within acceptable limits for non-sensitive applications.
- **User Experience**: The web-based interface provided an intuitive and user-friendly experience, allowing users to easily generate passwords and perform encryption and decryption operations.

### 6.2 Comparative Analysis

A comparative analysis was conducted between traditional alphanumeric passwords and Sanskrit consonant-based passwords in terms of security and usability. The results indicated that Sanskrit consonant-based passwords offer enhanced security due to their increased complexity and resistance to common attack vectors. However, the usability aspect was slightly compromised, as users may not be familiar with the Sanskrit script. To address this, the application provides a transliteration feature, allowing users to input passwords using the Latin alphabet, which are then converted to the corresponding Sanskrit consonants. This feature enhances usability while maintaining the security benefits of Sanskrit-based passwords.

## 7. CONCLUSION

This research demonstrates the feasibility and effectiveness of integrating Sanskrit consonant-based passwords into the DES encryption and decryption processes. By leveraging the rich linguistic heritage of Sanskrit, the proposed approach enhances password complexity and system security. The developed web-based application serves as a practical tool for users to generate secure passwords and perform encryption and decryption operations. Future work will focus on extending the approach to other cryptographic algorithms, such as AES, and exploring the integration of Sanskrit-based passwords into modern authentication systems. Additionally, research will be conducted to assess the usability and security of the system in real-world applications.

## REFERENCES

[1]. Chakraborty, N., & Mondal, S. (2015). A New Storage Optimized Honeyword Generation Approach for Enhancing Security and Usability. *arXiv*. Retrieved from https://arxiv.org/abs/1509.06094(arXiv)

[2]. Ghogare, P. (2018). Cryptography in Ancient India. *Academia.edu*. Retrieved from https://www.academia.edu/35239168/Cryptography_in_ancient_India.pdf(Academia)

[3]. Ghosal, P., Biswas, M., & Biswas, M. (2010). Hardware Implementation of TDES Crypto System with On Chip Verification in FPGA. *arXiv:1002.4836*. https://arxiv.org/abs/1002.4836

[4]. Schneier, B. (2015). Data Encryption Standard (DES). In *Applied Cryptography, Second Edition*. Wiley Online Library. https://doi.org/10.1002/9781119183471.ch12

[5]. Kaur, N., & Sodhi, S. (2016). Data Encryption Standard Algorithm (DES) for Secure Data Transmission. *International Conference on Advances in Emerging Technology*, ICAET2016, 2, 31–37. https://www.ijcaonline.org/proceedings/icaet2016/number2/25887-t036/

[6]. Alanazi, H. O., Zaidan, B. B., Zaidan, A. A., Jalab, H. A., Shabbir, M., & Al-Nabhani, Y. (2010). New Comparative Study Between DES, 3DES and AES within Nine Factors. *arXiv:1003.4085*. https://arxiv.org/abs/1003.4085

[7]. Abhishek Sachdeva. (2013). A Study of Encryption Algorithms AES, DES and RSA for Security. *Global Journal of Computer Science and Technology*, 13(E15), 32–40. Retrieved from https://computerresearch.org/index.php/computer/article/view/272(computerresearch.org)

[8]. Bruce Schneier. (2015). Data Encryption Standard (DES). In *Applied Cryptography, Second Edition*. Wiley Online Library. https://doi.org/10.1002/9781119183471.ch12(Wiley Online Library)

[9]. Nirmaljeet Kaur & Sukhman Sodhi. (2016). Data Encryption Standard Algorithm (DES) for Secure Data Transmission. *International Conference on Advances in Emerging Technology*, ICAET2016, 2, 31–37. https://www.ijcaonline.org/proceedings/icaet2016/number2/25887-t036/(IJCA)

[10]. Hamdan O. Alanazi, B. B. Zaidan, A. A. Zaidan, Hamid A. Jalab, M. Shabbir, & Y. Al-Nabhani. (2010). New Comparative Study Between DES, 3DES and AES within Nine Factors. *arXiv:1003.4085*. https://arxiv.org/abs/1003.4085(arXiv)

[11]. Prasun Ghosal, Malabika Biswas, & Manish Biswas. (2010). Hardware Implementation of TDES Crypto System with On Chip Verification in FPGA. *arXiv:1002.4836*. https://arxiv.org/abs/1002.4836(arXiv)

[12]. Lai, X., & Massey, J. (1991). International Data Encryption Algorithm. *Wikipedia*. https://en.wikipedia.org/wiki/International_Data_Encryption_Algorithm(Wikipedia).