Rapport sur le fonctionnement du code PHP implémentant l'algorithme RC4

Introduction

Le code fourni est une implémentation en PHP de l'algorithme de chiffrement RC4 (Rivest Cipher 4). L'algorithme RC4 est un chiffrement par flot, largement utilisé pour sa rapidité et sa simplicité.

Génération de clé (`generateKey`)

La fonction `generateKey` crée une clé basée sur la première lettre du message d'entrée. Si cette lettre est une consonne, la clé est construite en sélectionnant alternativement les caractères du message. Sinon, elle prend les caractères de manière alternative également, mais à partir des positions paires.

```
// Fonction pour générer une clé en fonction du message donné
function generateKey($message) {
    $firstLetter = strtolower($message[0]); // Récupère la première lettre du
message en minuscule
    $consonants = array('b', 'c', 'd', 'f', 'g', 'h', 'j', 'k', 'l', 'm', 'n',
'p', 'q', 'r', 's', 't', 'v', 'w', 'x', 'y', 'z');
    $key = '';

    // Vérifie si la première lettre est une consonne
    if (in_array($firstLetter, $consonants)) {
        // Utilise les lettres impaires comme clé
        for ($i = 0; $i < strlen($message); $i++) {
            if ($i % 2 !== 0) {
                $key .= $message[$i];
            }
        }
    } else {
        // Utilise les lettres paires comme clé
        for ($i = 0; $i < strlen($message); $i++) {
            if ($i % 2 === 0) {
                $key .= $message[$i];
            }
        }
    }

    return $key; // Retourne la clé générée
}
```

Algorithme RC4

Key Scheduling Algorithm (KSA)

La fonction `KSA` initialise un tableau d'état ($S) de 256 octets en fonction de la clé générée par `generateKey`. Elle utilise l'algorithme de génération d'horloge de RC4, mélangeant les octets du tableau d'état basé sur la clé fournie.

```
// Fonction pour initialiser l'état du tableau utilisé dans RC4
function KSA($key) {
 $S = range(0, 255);
 $j = 0;
 $key_length = strlen($key);

 for ($i = 0; $i < 256; $i++) {
     $j = ($j + $S[$i] + ord($key[$i % $key_length])) % 256;
     $temp = $S[$i];
     $S[$i] = $S[$j];
     $S[$j] = $temp;
 }

 return $S; // Retourne le tableau d'état initialisé
}
```

Pseudo-Random Generation Algorithm (PRGA)

La fonction `PRGA` est responsable de la génération de la séquence pseudo-aléatoire de la clé (keystream) basée sur l'état du tableau $S. Elle mélange les octets de manière itérative pour produire un flux de données pseudo-aléatoire.

```
// Fonction pour générer le flux de clés pseudo-aléatoire
function PRGA(&$S, $n) {
  $i = 0;
  $j = 0;
  $keystream = array();

  while (count($keystream) < $n) {
      $i = ($i + 1) % 256;
      $j = ($j + $S[$i]) % 256;
      $temp = $S[$i];
      $S[$i] = $S[$j];
      $S[$j] = $temp;
      $zi = $S[($S[$i] + $S[$j]) % 256];
      array_push($keystream, $zi);
  }

  return $keystream; // Retourne le flux de clés généré
}
```

Chiffrement RC4 (`chiffrement_RC4`)

Pour chiffrer un mot de passe, cette fonction génère d'abord une clé avec `generateKey`. Ensuite, elle crée le tableau d'état $S à l'aide de `KSA`. La séquence pseudo-aléatoire est produite par `PRGA` et combinée avec le mot de passe à l'aide d'une opération XOR pour produire le mot de passe chiffré en format hexadécimal.

```
// Fonction de chiffrement RC4
function chiffrement_RC4($password) {
  $key_bytes = generateKey($password);
  echo "Clé utilisée : ". $key_bytes ."\n";
  $password_bytes = $password;
  $S = KSA($key_bytes);
  $keystream = PRGA($S, strlen($password_bytes));
  $encrypted_password = '';

  foreach (str_split($password_bytes) as $index => $char) {
      $encrypted_password .= sprintf('%02x', ord($char) ^ $keystream[$index]);
  }

  return $encrypted_password; // Retourne le texte chiffré
}
```

Déchiffrement RC4 (`dechiffrement_RC4`)

La fonction `dechiffrement_RC4` prend un mot de passe chiffré et une clé. Elle utilise la clé pour générer le tableau d'état $S. Ensuite, elle utilise `PRGA` pour obtenir la séquence pseudo-aléatoire et effectue une opération inverse XOR pour retrouver le mot de passe d'origine.

```
// Fonction de déchiffrement RC4
function dechiffrement_RC4($encrypted_password, $key) {
  $key_bytes = $key;
  $S = KSA($key_bytes);
  $keystream = PRGA($S, strlen($encrypted_password) / 2);
  $encrypted_bytes = pack('H*', $encrypted_password);
  $decrypted_password = '';

  foreach (str_split($encrypted_bytes) as $index => $byte) {
      $decrypted_password .= chr(ord($byte) ^ $keystream[$index]);
  }

  return $decrypted_password; // Retourne le texte déchiffré
}
```

Conclusion

L'algorithme RC4 est un chiffrement par flot rapide et efficace. Ce code PHP met en œuvre les étapes essentielles de l'algorithme, de la génération de clé à la création du flux pseudo-aléatoire pour le

chiffrement et le déchiffrement. Cependant, il est important de noter que l'algorithme RC4 a montré des vulnérabilités et n'est plus recommandé pour une utilisation sécurisée.