http://vader-fr.fr/spip.php?article179

Fonctions sur les mots de passe

- Développement web - Les pages dynamiques avec PHP/MySQL - Exemples de fonctions PHP -

Date de mise en ligne : dimanche 1er juillet 2018

Diverses fonctions pour gérer l'authentification d'utilisateurs et le stockage sécurisé des mots de passe dans la BDD.

Vérification de la complexité d'un mot de passe

si la fonction renvoie une chaîne vide lorsqu'on l'appelle, tout baigne.

```
function chk_complex_mp($motpass){
 $message="";
 $patterns[0]="/^.*[A-Z].*+$/";        $complexe[0]="Il faut au moins une lettre majuscule";
 $patterns[1]="/^.*[a-z].*+$/";        $complexe[1]="Il faut au moins une lettre minuscule";
 $patterns[2]="/^.*[0-9].*+$/";        $complexe[2]="Il faut au moins un chiffre";
 $patterns[3]="/^.{8,}$/";               $complexe[3]="Il faut au moins 8 caractères";
 for ($i=0;$i
 $pattern=$patterns[$i];
 if (!preg_match("$pattern",$motpass)){
 $message.=$complexe[$i];
 }
 }
 return $message;
 }
```

Hashage de mot de passe avec sel, par algorithme choisi

Cette fonction n'est qu'un exemple de fonction de cryptage, il est possible de faire plus souple/plus complexe.
A partir d'un mot de passe en clair, d'une chaîne de salage et d'un algorithme de hashage, elle :

- casse le sel en 3 parties de 10 caractères
- casse le mot de passe en 3 parties égales
- hash une première fois les morceaux de mot de passe mélangés aux morceaux de sel
- re-casse le mot de passe salé et hashé en 3 parties égales
- et enfin re-mélange le sel avec le mot de passe hashé.

```
function genere_motpasse($motpasse,$alt,$algo){
 $alt1=substr($alt, 0, 10);
 $alt2=substr($alt, 10, 10);
 $alt3=substr($alt, 20, 10);
```

```
$taillepass=strlen($motpasse);
$taillemorceau=intval($taillepass/3);
$pass1=substr($motpasse,0,$taillemorceau);
$pass2=substr($motpasse,$taillemorceau,$taillemorceau);
$pass3=substr($motpasse,($taillemorceau*2));
$passfin=hash($algo,"$pass1$alt1$pass2$alt2$pass3$alt3");
$taillepassf=strlen($passfin);
$taillemorceauf=intval($taillepassf/3);
$passf1=substr($passfin,0,$taillemorceauf);
$passf2=substr($passfin,$taillemorceauf,$taillemorceauf);
$passf3=substr($passfin,($taillemorceauf*2));
$passfou="$passf1$alt1$passf2$alt2$passf3$alt3";
// le mot de passe coupé, salé, hashé et re-salé est :
return $passfou;
}
```

Génération de mot de passe aléatoire et/ou de sel

voir l'article sur le salage en informatique sur Wikipedia.

Cette fonction génère donc une chaîne aléatoire, par exemple pour le sel aléatoire, qui devra être stocké, éventuellement pour chaque utilisateur.
Cela permet de se protéger (dans une certaine mesure) des attaques par rainbow table.

```
// génère une chaîne alpha-numérique aléatoire de longueur $taille
function genere_chaine_aleatoire($taille) {
$code_aleatoire="";
$chaine="ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
srand((double)microtime()*1000000);
for($i=0; $i<$taille; $i++) {
$code_aleatoire.=$chaine[rand()%strlen($chaine)];
// ou $code_aleatoire .= substr($chaine,rand()%(strlen($chaine)),1);
// ou $code_aleatoire .= $chaine[ rand(0, ($nb_chars-1)) ];
}
// ou $code_aleatoire = str_shuffle('ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789');
return $code_aleatoire;
}
```

si l'on veut casser et mélanger le sel et le mot de passe avant/après cryptage selon les chiffres compris dans le sel, il est possible, à sa génération, de rajouter une chaîne "0 à 9" au sel si celui-ci ne contient pas de chiffre, puis de mélanger le sel par la fonction str_shuffle().

Choix d'un algorithme de hashage

Ce bout de code cherche dans le tableau renvoyé par la fonction "hash_algos" les meilleurs algorithmes de hashage.
hash_algos() renvoie les algorithmes supportés sur le système.

Une fois trouvé, l'algorithme à utiliser pourra être stocké soit dans le fichier connect.php, soit dans la base de données, dans une table de méta-variables.
Les algorithmes à chercher sont triés par ordre "de préférence" du plus sûr (sha512, whrilpool...) au moins sûr (crc32).

```
// chercher dans hash_algos si sha512, sha256, sha1... sont utilisables.
$algos_prefs=array("sha512", "whirlpool", "sha384", "ripemd320", "sha256", "snefru256", "ripemd256",
"snefru", "gost", "sha224", "ripemd160", "ripemd128", "fnv164", "joaat", "crc32b", "adler32", "fnv132",
"crc32");
$algo=null;
$i=0;
$nbalgos=sizeof($algos_prefs);
while (($i<$nbalgos)&&($algo==null)){
if (in_array($algos_prefs[$i],hash_algos())){
$info.="".$algos_prefs[$i]." est supporté";
$algo=$algos_prefs[$i];
}else{
$info.="".$algos_prefs[$i]." non supporté";
$i++;
}
}
if ($algo==null){
$info.="Aucun algorithme supporté parmi ceux cherchés";
$debug.="Aucun algorithme supporté parmi ceux cherchés";
}
```