基于 PHP 数据加密安全性探讨

刘凯凌

（广东省清远市技师学院，广东 清远 511515）

摘　要：基于 PHP 的数据加密技术，通过多种加密方法应对不同情况下的数据安全需求，为保障用户的数据安全以及网站平台搭建和发展提供了必要的支持。

关键词：PHP 数据加密；数据安全性

DOI:10.16184/j.cnki.comprg.2017.07.034

随着计算机技术的不断发展，网站的搭建已经进入到了更高的层次，运用多种程序进行组合的 LAMP，即 Linux+Apache+Mysql+PHP，成为当前最为流行的技术。其中 PHP 作为一种简单高效且兼容性较强的对象脚本语言，一直在行业内部广受好评，成为网站搭建的主要技术之一。它经常用于制作动态网页，负责网站和用户之间的信息交互。然而在上传或者下载文件的过程中，有一部分文件需要进行加密处理，这就要求 PHP 对文件进行加密，以保证数据的安全性。通过分析几种 PHP 文件加密的方法，对其进行初步比较，以期能为广大用户在使用和加密文件的时候提供一些便利。

1　PHP 加密函数

PHP 的加密函数有 crypt()、md5() 和 sha1() 这 3 种，其中 crypt() 用于单向加密，所谓的单向加密就是将需要加密的内容进行加密之后，无法将密文转换成为可读的内容，因此单向加密的应用范围较狭窄，一般用于用户名认证和密码输入等情况；当用户进入系统时，只需要将密文口令输入，经过系统验证与存储的口令一致，即可通过。该函数的具体格式为 crypt()：string crypt (string input，string[，string salt]) 其中，input、string 是指用户输入需要加密的字符串的位置，salt 则是为了增加被加密后的字符串数目，以增加安全度；即使用户没有输入 salt 参数，在调用该函数时系统也会随机生成。例如输入口令为 hello world，那么在函数中：

```
<?php
echo“Standard DES：“.crypt(“hello world”).”\n<br/>”;
?>
```

则显示出来为 Standard DES：$l $r35.Y52. $iyiFuvM. zFGsscpUOaz4e.这样即使在输入的时候被人看到或者复制，也不会对口令安全造成威胁。

md5()和 sha1()属于哈希算法，它是不可逆的一种算法，通过截取任意一段的初始信息，将其进行转换，所得到的内容就是哈希值，且长度固定。这样即使信息丢失，对哈希值进行分析也是无意义的，因为它与原来的信息并无直接联系，因此具备较强的加密功能。md5() 使用了 MD5 散列算法，将一个长度不固定的信息转换为 128 位的信息摘要，该函数的具体格式为 md5 (string，raw)，其中 String 指用户输入的需要转换的字符串位置，raw 则是规定输出格式为二进制或十六进制。仍然以 hello world 转换口令，那么在函数中：

```
<?php
$string=“Hello，PHP world!　”；
$out=md5($string)；
print“输出：$out”；
?>
```

则显示结果为：7996b5e0804042fcl531907a4900f190e

sha1() 函数使用了 SHA-1 的散列算法，其原理与 md5() 类似。md5() 和 sha1() 经常用于验证信息的完整性，即通过计算文件的哈希值来验证文件是否被修改，因此在 PHP 中还需要两个函数来对哈希值进行计算，md5_file() 和 sha1_file()，一旦发现计算出的哈希值与原始值不同，就可以判断文件遭到了修改。这 3 种函数虽然操作简单，但是都是不可逆的，无法对密文进行解读；但是在更高层次的加密工作中，这些简单的加密函数的加密效果就显得有些捉襟见肘了。

2　Pear 类库和 PHP 的扩展

在高级加密工作中，对加密函数提出了更高的要求，因此出现了 Pear 类库和 PHP 的扩展。它们具有十分强大的加密功能，丰富多样的函数种类提供了各种加密支持，以满足不同用户的加密需求。其中 MCrypt 就是典型的高级加密函数，它支持大多数常见的加密算法，使加密和解密工作十分便捷；MHash 支持 hash 算法，例如 SHA、MD5 和CRC 等，可以通过计算来获得哈希值；Crypt_Blowfish，从名字就可以看出，它是 Blowfish 算法的高级版本，能够支持双向加密，既加密初始信息之后，还可以将其转换为可读内容。

作者简介：刘凯凌（1977-），男，讲师，研究方向：网站设计规划以及网络编程、计算机软件应用等教学。

收稿日期：2017-01-13

Crypt_RSA 也是同样的类型，是 RSA 算法的高级版本，它不仅可以进行双向加密，且对密钥的长度没有限制。Crypt_HMAC 易于操作，使用简便，只需通过密钥，散列计算方法和明文就可以计算散列值，且支持 MD5 和 SHA1 算法。Crypt_DiffieHellman 主要用于在 PHP.5 版本下实现密钥交换协议，即双方通过该协议产生一个密钥，然后双方在持有该密钥的情况下可以在非安全的环境下通信，且不会受到信息泄露的威胁。这些扩展和类库的功能各有侧重，用户可以根据实际的加密需求来选择适合的功能。目前来看，MCrypt 是 PHP 扩展中最为常见的一种加密方法，因为它支持的算法种类繁多，在实际使用的时候能够提供更多的便利。但是 PHP 的扩展和类库也有缺点，那就是用户的密钥管理，尽管一些网站采取了加密登录口令的方式，其效果仍然不甚理想。

3 在 PHP 中使用 GnuPG 软件

GnuPG，又称 GPG，是基于 PGP 机制的加密软件，它能够为网络用户的数据传输提供很好的保护。所谓的 PGP 机制，其实质就是一款邮件加密软件，它使用 RSA 公共密钥加密体系来对用户数据进行加密，并通过增加数字签名以使收件人确认邮件发出地址是否正确。这样既保障了通讯安全，也免去了传递密钥的步骤。PGP 最大的优点就是对于密钥的管理，它将 RSA 和传统加密进行结合，即运用加密函数将数字签名的信息截取并加密为固定长度的摘要，使得安全性能大大提高。通常情况下，GPG 和 PGP 是相同的，但是 PGP 使用了 IDEA 的专利算法，所以使用的时候需要经过认证；而 GPG 是基于 PGP 机制的再开发软件，不涉及专利算法的使用，用户在使用时也就相对便利了很多。GPG 使用的加密算法是非对称的：用户会得到两个密钥，公钥和私钥各一个。公钥是公开的，便于对方与自己进行数据传输时使用，而私钥并不公开，由用户自己保存，这样私钥可以解密经过公钥加密的数据。假设有用户 a 要发送一个文件给用户 b，首先要使用 b 在服务器上的公钥对文件进行加密，然后再发送；用户 b 接受到文件之后，再用自己的私钥解密文件，从而实现了加密过程。

想要在 PHP 中使用 GPG 其实很简单，只要通过 shell_exec() 函数来进行调用就可以。该函数的作用是在PHP 脚本中通过 shell 调用外部命令，使得所有 GPG 命令都能在 PHP 脚本中被灵活使用，就如同在 PHP 中产生了合适的运行环境一样。

```
<?php
$gpg='/usr/bin/gpg';//指出 gpg 的位置
putenv("HOME=/yourpath/");
putenv("GNUPGHOME=/yourpath/.gnupg");
$file_dir="/yourfilepath/";
$file_name="public_file.doc";
$recipient=$_SESSION['userinfo']['email'];
$secret_file=$file_dir.$file_name;
 $cmdgpg ="$gpg—quiet?no - secmem –warning—r
$recipient—
e$secret_file";
shell_exec($cmdgpg);
$cmdrm="rm". $file_dir.$file_name; //如有需要可以
将服务器端的明文删除
exec($cmdrm);
$file_name_gpg="public_file.doc.gpg";//下载密文
 $url ="http://". $_ SERVER ['HTTP_HOST']. $file_dir.
$file__
name_gpg;
Header("location:".$url);
<?php
```

该程序就是在 PHP 中使用 GPG 进行加密的过程。

4 各种方法的比较

加密函数的优点是便于操作，简便易懂，且基本不会受到系统版本的影响；缺点则是支持的算法种类较少，且都是单向加密，用户无法进行解密。这对于数据传输来说并不适合，只能应用于身份验证和登录口令的加密。扩展和类库对于加密算法的支持较好，能够适应大部分的算法，使用户拥有了充分的选择，而且扩展和类库具有多种不同功能的加密函数，能够有针对性地满足用户需求。但是它缺乏科学的密钥管理机制，难以解决通信工作中的加密解密需求。GPG 相对于前面两者来说，是更加专业的加密软件，它功能齐全并且可以增加数字签名以提高安全性；它所采用的非对称加密算法既确保了数据的安全，也完善了密钥管理机制：公钥和私钥的搭配很好地解决了通信中对文件加密和解密的需求，保障了数据的安全。但是它也有一些缺点，那就是对用户的操作技巧要求较高，用户必须熟悉基本的 GPG 软件功能，并对密钥的生成和使用有所了解。

5 结语

所讨论的 3 种在 PHP 下对文件进行加密的技术都是为了保证数据在网络中进行传输时不会出现泄漏或者丢失的情况。这 3 种方法各有所长，可以独立使用也可以搭配运用，开发人员在进行选择的时候要充分考虑到用户的需求以及实际运行的环境，这样开发出来的加密技术才能最大限度地满足用户和网站的需求。

参考文献

[1] 杨明华. LAMP 网站开发黄金组合 Linux+Apache+MySQL+PHP [M]. 北京电子工业出版社，2008.

[2] Andi Gutmans, PHP5 权威编程【M】. 简张桂，译. 北京电子工业出版社，2007.