云盘系统总结

一、请总结你的所有已完成功能点

- 基于网页的用户注册与登录系统

  1、使用https绑定证书到域名

  2、允许用户注册到系统

  ```
  用户名的合法字符集范围：中文、英文字母、数字
  用户口令长度限制在36个字符之内
  对用户输入的口令进行强度校验，禁止使用弱口令
  ```

  3、使用合法用户名和口令登录系统

  4、对用户口令进行慢速哈希后存储，存储的口令即使被公开，也无法还原/解码出原始明文口令

  5、找回密码功能（邮件找回，重置密码）

  6、微博和支付宝的OAuth授权登录

- 基于网页的文件上传加密与数字签名系统

  文件上传，对上传文件进行处理

  ```
  限制文件大小：< 10MB
  限制文件类型：office文档、常见图片类型（.jpg,.png,.gif,.doc,.dox）
  匿名用户禁止上传文件
  对文件进行对称加密存储到文件系统
  系统对加密后文件进行数字签名
  ```

- 基于网页的加密文件下载与解密

  1、匿名用户可以下载加密后的文件和其关联的数字签名文件

  ```
  客户端对下载后的文件进行数字签名验证（签名验证专栏）
  客户端对下载后的文件可以解密还原到原始文件（文件解密专栏）
  ```

  2、已登录用户可以在自己的主页中，下载自己之前上传文件的原文件

3、文件分享链接的随机生成（24小时内有效，有效访问次数5次）

4、下载文件可以本地校验文件完整性

二、X.509证书中各个字段含义、用途解释说明

参考：https://segmentfault.com/a/1190000002568019

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=CN, ST=beijing, L=beijing, O=cuc, OU=cuc,
CN=www.enjoycryptology.com
        Validity
            Not Before: Jul 19 03:03:28 2017 GMT
            Not After : Jul 19 03:03:28 2018 GMT
        Subject: C=CN, ST=beijing, O=cuc, OU=cuc, CN=www.enjoycryptology.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    [illegible]
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                B4:40:F1:3B:46:52:84:D1:83:CE:08:8A:28:AC:1D:42:70:35:B2:33
            X509v3 Authority Key Identifier:

keyid:24:28:D1:0C:A4:AA:8A:AC:39:5B:BC:BD:43:FD:CB:53:DF:5A:FE:66

    Signature Algorithm: sha256WithRSAEncryption
         5b:89:b6:9d:44:87:4e:16:0b:af:b1:c9:b2:b7:13:24:41:00:
-----BEGIN CERTIFICATE-----
MIIDwjCCAqqgAwIBAgIBATANBgkqhkiG9w0BAQsFADBvMQswCQYDVQQGEwJDTjEQ
MA4GA1UECAwHYmVpamluZzEQMA4GA1UEBwwHYmVpamluZzEMMAoGA1UECgwDY3Vj
MQwwCgYDVQQLDANjdWMxIDAeBgNVBAMMF3d3dy5lbmpveWNyeXB0b2xvZ3kuY29t
```

X.509 v3证书基本证书字段

1. 证书版本号(Version)
版本号指明X.509证书的格式版本，现在的值可以为:
    1) 0: v1
    2) 1: v2
    3) 2: v3
version3相比较version1增加了用户标识、签发者标识和扩展字段

2. 证书序列号(Serial Number)
序列号指定由CA分配给证书的唯一的"数字型标识符"。当证书被取消时，实际上是将此证书的序列号放入由CA签发的CRL(证书作废列表)中，这也是序列号唯一的原因。

3. 签名算法标识符(Signature Algorithm)
签名算法标识用来指定由CA签发证书时所使用的"签名算法"。算法标识符用来指定CA签发证书时所使用的:
    1) 公开密钥算法
    2) hash算法
example: sha256WithRSAEncryption
须向国际知名标准组织(如ISO)注册

4. 签发机构名(Issuer)
此域用来标识签发证书的CA的X.500 DN(DN-Distinguished Name)名字。包括:
    1) 国家(C)
    2) 省市(ST)
    3) 地区(L)
    4) 组织机构(O)
    5) 单位部门(OU)
    6) 通用名(CN)
    7) 邮箱地址

5. 有效期(Validity)
指定证书的有效期，包括:
    1) 证书开始生效的日期时间
    2) 证书失效的日期和时间
每次使用证书时，需要检查证书是否在有效期内。

6. 证书用户名(Subject)
指定证书持有者的X.500唯一名字。包括:
    1) 国家(C)

2) 省市(ST)
3) 地区(L)
4) 组织机构(O)
5) 单位部门(OU)
6) 通用名(CN)
7) 邮箱地址

7. 证书持有者公开密钥信息(Subject Public Key Info)
证书持有者公开密钥信息域包含两个重要信息:
1) 证书持有者的公开密钥的值
2) 公开密钥使用的算法标识符。此标识符包含公开密钥算法和hash算法。

8. 扩展项(extension)
X.509 V3证书是在v2的基础上一标准形式或普通形式增加了扩展项，以使证书能够附带额外信息。标准扩展是指
由X.509 V3版本定义的对V2版本增加的具有广泛应用前景的扩展项，任何人都可以向一些权威机构，如ISO，来
注册一些其他扩展，如果这些扩展项应用广泛，也许以后会成为标准扩展项。

9. 签发者唯一标识符(Issuer Unique Identifier)
签发者唯一标识符在第2版加入证书定义中。此域用在当同一个X.500名字用于多个认证机构时，用一比特字符串
来唯一标识签发者的X.500名字。可选。

10. 证书持有者唯一标识符(Subject Unique Identifier)
持有证书者唯一标识符在第2版的标准中加入X.509证书定义。此域用在当同一个X.500名字用于多个证书持有者时，
用一比特字符串来唯一标识证书持有者的X.500名字。可选。

11. 签名算法(Signature Algorithm)
证书签发机构对证书上述内容的签名算法
example: sha256WithRSAEncryption

12. 签名值(Issuer's Signature)
证书签发机构对证书上述内容的签名值

三、WEB服务器使用的证书和CA使用的证书有什么区别和联系？

> 服务器证书(server certificates) 组成Web服务器的SSL安全功能的唯一的数字标识。通过相互信任的第三方组织获得，并为用户，提供验证Web站点身份的手段。这意味着服务器证书确保用户关于web服务器内容的验证，同时意味着建立的HTTP连接是安全的。当我们通过 HTTPS 访问页面时，浏览器会主动验证web服务器证书信息是否匹配，也会验证证书是否有效。

> WEB服务器证书是由CA签发的，它的可靠性可以通过CA证书进行验证。CA的作用就是提供服务器证书，加强服务端和客户端之间信息交互的安全性，以及证书运维相关服务。网上的公众用户通过验证 CA 的签字从而信任 CA ，任何人都可以得到 CA 的证书（含公钥），用以验证它所签发的证书。

四、简述你的口令安全存储策略。

- 用户口令存储策略：

  用户口令经过慢速哈希存储在数据库。

  慢速哈希通过增加哈希计算的时间，可以有效地抵御暴力穷举攻击。我们使用了php自带的哈希函数：

  ```
  string password_hash ( string $password , integer $algo [, array $options ] )
  ```

  此函数默认使用bcrypt算法进行哈希计算。bycryt算法是基于blowfish算法的改进版，是慢速哈希计算的优秀算法。

- 文件加密密码存储策略：

  文件加密密码通过服务器公钥加密之后存储在数据库。

  文件加密密码的加密主要是为了保证数据库中存储的不是裸数据，防止因数据库的安全问题导致文件加密密码的泄露。

  公钥加密的使用php函数：

  ```
  bool openssl_public_encrypt ( string $data , string &$crypted , mixed $key [, int $padding = OPENSSL_PKCS1_PADDING ] )
  ```

  此函数使用RSA算法进行公钥加密，同时默认是用了文件填充方式PKCS1_PADDING。

五、你是如何实现弱口令检测的？

前后端都通过对用户输入的口令进行正则表达式的匹配，从而对该口令进行评分，评分为0或1即判断为弱口令，评分为2为中等口令，评分为3则为强口令。
相关评分规则如下：

```
  //初始评分设为0级
  if(preg_match('/(?=.{6,}).*/',$password)==0)//6位以下密码统一为弱密码
  {
                $score = 1;
  }

  else if(preg_match('/^(?=.{8,})(?=.*[A-Z])(?=.*[a-z])(?=.*[0-9])(?
=.*\\W).*$/',$password))//8位以上且包含大小写字母、数字、特殊字符为强密码
  {
                $score = 3;
  }
  else if(preg_match('/^(?=.{7,})(((?=.*[A-Z])(?=.*[a-z]))|((?=.*[A-
Z])(?=.*[0-9]))|((?=.*[a-z])(?=.*[0-9]))).*$/',$password))//7位以上且
字母、数字、字符任意包含两项为中密码
  {
                $score = 2;
  }
  else//6位以上但只有单一字符集的为弱密码
 {
                $score = 1;
 }
```

六、你是如何实现安全的文件上传的？

文件通过https加密传输传输。到达服务器后，文件通过对称加密之后存储。

对称密码随机生成，使用函数：

```
string openssl_random_pseudo_bytes ( int $length [, bool
&$crypto_strong ] )
```

此函数生成指定长度的随机字节，再通过base64处理变化成可读的加密密码。

文件加密算法使用的是256位的AES，采取的CBC模式。初始变量采用相同的生成随机字节的方式生成，初始变量可以公开存储。加密使用的函数是：

```
string openssl_encrypt ( string $data , string $method , string $key
[, int $options = 0 [, string $iv = "" [, string &$tag = NULL [,
string $aad = "" [, int $tag_length = 16 ]]]]] )
```

保存原始文件哈希值，加密后文件哈希值以供校验。

请展示并说明你的

1、文件加密代码片段

```
$fileName = $path.$uniName;
$file = fopen($fileName, "r+") or die("Unable to open file!");
$data= fread($file,filesize($fileName));
fseek($file,0);
$method = "AES-256-CBC";
$iv =openssl_random_pseudo_bytes(16);
$enbase64iv = base64_encode($iv);
$bytes = openssl_random_pseudo_bytes(10);
$key= substr(str_replace(['/', '+', '='], '', base64_encode($bytes)), 0, 8);
$code = openssl_encrypt($data,$method,$key,OPENSSL_RAW_DATA,$iv);
$encryfilemd5 = md5($code);
fwrite($file, $code);
fclose($file);
$publickey = openssl_pkey_get_public(file_get_contents('../public.key'));
if(!openssl_public_encrypt ($key ,$encryptedkey,$publickey)){
   echo "public encrypt error!!!";
```

2、文件解密代码片段

```
if($iv == '')
{
  echo "error!!!!";
  $content = openssl_decrypt($data,$method,$filekey);
}
else {
  echo $iv;
  $content = openssl_decrypt($data,$method,$filekey,OPENSSL_RAW_DATA,$iv);
}
$uniName=md5_file($tmp_name);
$file = fopen($tmp_name, "r+") or die("Unable to open file!");
$data= fread($file,filesize($tmp_name));
fclose($file);
```

3、文件签名代码片段

```php
<?php
function signFile($file,$out,$username)
{
  if(!is_file($file)) {
      echo "$file does not exist!\n";
      exit(1);
  }


  $username_hash=md5($username);

  $data = file_get_contents($file);
  $priv_key = openssl_pkey_get_private("file:///keys/".$username_hash.".key");

  //create signature
  openssl_sign($data, $signature, $priv_key, OPENSSL_ALGO_SHA256);
  $signature=$signature.$username_hash;
  file_put_contents($out, $signature);
  return;
}
```

通过传进函数的username确定用户的私钥信息，通过openssl_pkey_get_private获得私钥，使用 ` openssl_sign($data, $signature, $priv_key, OPENSSL_ALGO_SHA256);` 将加密后的文件内容$data,使用私钥$priv_key，采用SHA256算法进行签名，得到签名$signature。

4、文件签名验证代码片段

```php
<?php
function verify($file,$sig)
{
  If(!is_file($file)) {
    echo "<script>alert('data". $file." does not exist!');location='../html/verifysign.php';</script>";
    exit(1);
  }
  If(!is_file($sig)) {
    echo "<script>alert('signature file". $file." does not exist!');location='../html/verifysign.php';</script>";
    exit(1);
  }
  $data = file_get_contents($file);
  $file_sign = fopen($sig,"r");
  $signature=fread($file_sign,"256");
  fseek($file_sign,"256");
  $username_hash=fread($file_sign,"32");
  $pub_key  = openssl_pkey_get_public("file:///keys/".$username_hash.".crt");
  fclose($file_sign);
  $ok = openssl_verify($data, $signature, $pub_key, OPENSSL_ALGO_SHA256);
  If ($ok == 1) {
    echo "<script>alert('valid!');location='../html/verifysign.php';</script>",PHP_EOL;
  } elseif ($ok == 0) {
    echo "<script>alert('invalid!');location='../html/verifysign.php';</script>",PHP_EOL;
  } else {
    echo "<script>alert('error ! !".openssl_error_string()."');location='../html/verifysign.php';</script>";
  }
}
```

打开签名文件，读取前256个字节为系统对加密文件的签名，之后32个字节为上传用户的用户名的md5值，通过该md5值获取到该用户公钥的path，然后使用openssl_pkey_get_public获取的公钥， `openssl_verify($data, $signature, $pub_key, OPENSSL_ALGO_SHA256);`使用公钥对签名文件进行验证，从而验证用户身份真实性和内容完整性。

5、文件完整性验证代码片段 (S)

```php
$file=$_FILES['filename'];
$hash=$_POST['hash'];
$tmp_name=$file["tmp_name"];
// default md5
$hashtmp = md5_file($tmp_name);
if($hash == $hashtmp)
{
  echo "验证成功，文件完整";
}
else {
  echo "验证失败，文件不完整";
}
```

七、同一个用户的不同文件是否使用相同的对称加密密钥？如果是，请说明其中存在的安全风险。如果否，请结合代码展示你的文件对称加密密钥的存储和提取使用策略

同一用户的不同文件使用不同的对称加密密钥。

```
$bytes = openssl_random_pseudo_bytes(10);
$key= substr(str_replace(['/', '+', '='], '', base64_encode($bytes)), 0, 8);
```

当已登录用户下载文件时需要将加密文件进行解密

```
$rawkey = base64_decode($row['key']);
$privatekey = openssl_pkey_get_private(file_get_contents(PrivatePath));
openssl_private_decrypt ($rawkey,$filekey ,$privatekey);
```

从数据库中取出加密后的文件密码，使用数据库私钥进行解密之后使用。

八、你的文件下载过期策略是如何设计并实现的？

构造下载分享链接：

```
$token_sql=base64_encode(openssl_random_pseudo_bytes(8));
$token=base64_encode($username.$token_sql);
$url = "http://www.enjoycryptology.com/summer_item/php/fileshare.php?filename=".
$filename."&token=".$token;

$requesttime = time();
$times=5;
$sql = "insert FileShare values('".$username."','".$filename."','".$token_sql."','".$times."','".
$requesttime."')";
$result = $mysqli->query($sql);
```

使用openssl_random_pseudo_bytes(8)随机产生一个8位字节流，通过base64加密变为可读字符，而后与username连接再进行base64加密，构造出下载链接分享唯一标识符。将请求时间和允许请求的次数存入数据库，每访问一次数据库中次数减一。

链接有效性验证：

```
if($username == "" && $token == "")
{
 $msg = '该链接无效！';
 echo "<script>alert('".$msg."');location='../html/index.php';</script>";
 exit();

}
else
{
  if(time()-$requesttime>24*60*60)//链接24小时有效
  {
    $msg = '该链接已过期！';
    echo "<script>alert('".$msg."');location='../html/index.php';</script>";
    exit();
  }
  else if ($times==0)
  {
    $msg = '该链接分享次数已达上限！';
    echo "<script>alert('".$msg."');location='../html/index.php';</script>";
    exit();
  }
```

获取分享链接中的`filename`和`token`，去数据库中进行匹配，若匹配到分享记录，则验证当前时间与分享时间的差值，若大于规定时间，则证明该请求已过期，若数据库中请求次数减为`0`次，则该链接分享次数已达上限。

九、常见对称加密工作模式有哪些？各自应用场景、优缺点说明。
（ECB\CBC\CFB\OFB）

- ECB模式（密码本模式）

  优点：简单、有利于并行计算、无错误传播
  缺点：明文分组重复时，密文也重复，容易统计分析攻击；结构化数据，容易产生大量重复的密文；无法隐藏原明文数据的模式
  应用场景：单个数据的安全传输，短数据的加密

- CBC模式（密文分组链接模式模式）

  优点：安全性好于ECB,适合传输长度长的报文
  缺点：不利于并行计算；需要初始化向量IV；存在密文错误传播
  应用场景：普通的面向分组的传输认证

- CFB模式（密文反馈模式)

  优点：使用于数据以比特/字节为单位的场合；隐藏了明文模式；可以将分组密码转化为流模式
  缺点：不利于并行计算；存在明密文错误传播；容易受到重放攻击
  应用场景：普通的面向分组的传输认证

- OFB模式（输出反馈模式）

  优点：传输过程中的比特错误不会被传播
  缺点：易受到对消息流得篡改攻击
  应用场景：噪声频道上的数据流的传输（卫星通信）

十、简述RSA加密算法和RSA签名算法之间的关系？

> 数据传输双方使用RSA算法各生成属于自己的一对公私钥，RSA加密算法是发送方使用接收方的公钥对加密数据的对称密钥加密后发送给接收方，而后接收方用自己的私钥解密即得到对称密钥，这里RSA加密用来传递密钥。RSA签名算法是用来进行身份验证和完整性验证的，发送方用自己的私钥对经过哈希值计算的数据进行加密，接收方能够通过发送方公开的公钥进行解密，从而验证发送方身份的真实性。

十一、通过PHP实现文件散列值计算有哪些方法？

```
md2          md4          md5          sha1         sha224       sha256
sha384       sha512       ripemd128    ripemd160    ripemd256    ripemd320
whirlpool    tiger128,3   tiger160,3   tiger192,3   tiger128,4   tiger160,4
tiger192,4   snefru       snefru256    gost         gost-crypto  adler32
crc32        crc32b       fnv132       fnv1a32      fnv164       fnv1a64
joaat        haval128,3   haval160,3   haval192,3   haval224,3   haval256,3
haval128,4   haval160,4   haval192,4   haval224,4   haval256,4   haval128,5
haval160,5   haval192,5   haval224,5   haval256,5
```

PHP支持的所有的哈希算法，在文件哈希时并没有特殊说明，所以都可以使用。

十二、你是如何实现匿名用户禁止上传文件功能的？

> 采用session机制。先通过session_start();开启一个session会话，系统为每一个开启了 session 会话的访问者建立一个唯一的会话 ID ，用于识别用户。对应的具体session存储在服务器端。

具体验证函数：

```
function Judgelog(){
    <?php
if($_SESSION['username'] == null){
    ?>

    alert("匿名用户禁止上传文件")
    return false

    <?php
 }else
    {
    ?>
    return true

        <?php
        }
        ?>
    }
```

每次登陆成功，对应session[id]会记录下登陆者用户名：`$_SESSION['username'] = $_POST['username']; `，但点击上传按钮时，对应函数会判断$_SESSION['username']是否为空，为空则证明使用者未登录系统，弹框提醒用户禁止文件上传。

十三、请展示并说明你的数据库表结构设计

- 表一：Users——存储已注册用户信息的数据库

```
mysql> desc Users;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| id          | int(11)      | NO   | PRI | NULL    | auto_increment |
| username    | varchar(100) | NO   |     | NULL    |                |
| password    | varchar(100) | YES  |     | NULL    |                |
| Email       | varchar(100) | YES  |     | NULL    |                |
| phonenumber | varchar(50)  | YES  |     | NULL    |                |
| getpasstime | int(10)      | YES  |     | NULL    |                |
| weiboid     | varchar(100) | YES  |     | NULL    |                |
| alipayid    | varchar(100) | YES  |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
```

字段说明：

id：用户编号，自动增长
username：用户名
password：用户加密后的口令
Email：注册时用户使用的邮箱
phonenumber：注册时用户使用的手机号码
getpasstime：用户申请找回密码时记录的时间
weiboid：第三方授权登录时存储的唯一标识（weibo授权）
alipayid：第三方授权登录时存储的唯一标识（支付宝授权）

- 表二：FileKey——存储上传文件信息的数据库

```
mysql> desc FileKey;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| username     | varchar(100) | YES  |     | NULL    |       |
| filename     | varchar(100) | NO   |     | NULL    |       |
| uniname      | varchar(100) | NO   |     | NULL    |       |
| size         | varchar(20)  | NO   |     | NULL    |       |
| uploadtime   | varchar(50)  | NO   |     | NULL    |       |
| key          | varchar(500) | NO   |     | NULL    |       |
| cert         | varchar(500) | NO   |     | NULL    |       |
| iv           | varchar(100) | YES  |     | NULL    |       |
| encryFileMd5 | varchar(100) | YES  |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
```

字段说明：

username：文件上传者的用户名
filename：上传文件原名
uniname： 上传文件另设的唯一文件名，防止在目录下因为文件名相同而产生覆盖
$uniName=md5(uniqid(microtime(true),true)).$ext;//md5加密，uniqid产生唯一id，microtime做前缀
size：上传文件的大小
uploadtime：上传文件时间
key：用于加密的对称密钥
cert：原打算存储用户公钥证书信息，现更改为直接存储为文件。
iv：用于加密的初始向量iv
encryFileMD5：上传文件的散列值

- 表三：FileShare——存储被分享文件信息的数据库

```
mysql> desc FileShare;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| username    | varchar(100) | YES  |     | NULL    |       |
| filename    | varchar(100) | NO   |     | NULL    |       |
| token       | varchar(100) | NO   |     | NULL    |       |
| times       | int(50)      | NO   |     | NULL    |       |
| requesttime | int(10)      | NO   |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
5 rows in set (0.04 sec)
```

字段说明：

username：文件上传者的用户名
filename：上传文件原名
token：生成的分享链接的唯一标识符
times：分享次数，默认最多为5次
requesttime：请求分享的时间，用来之后判断请求链接是否过期