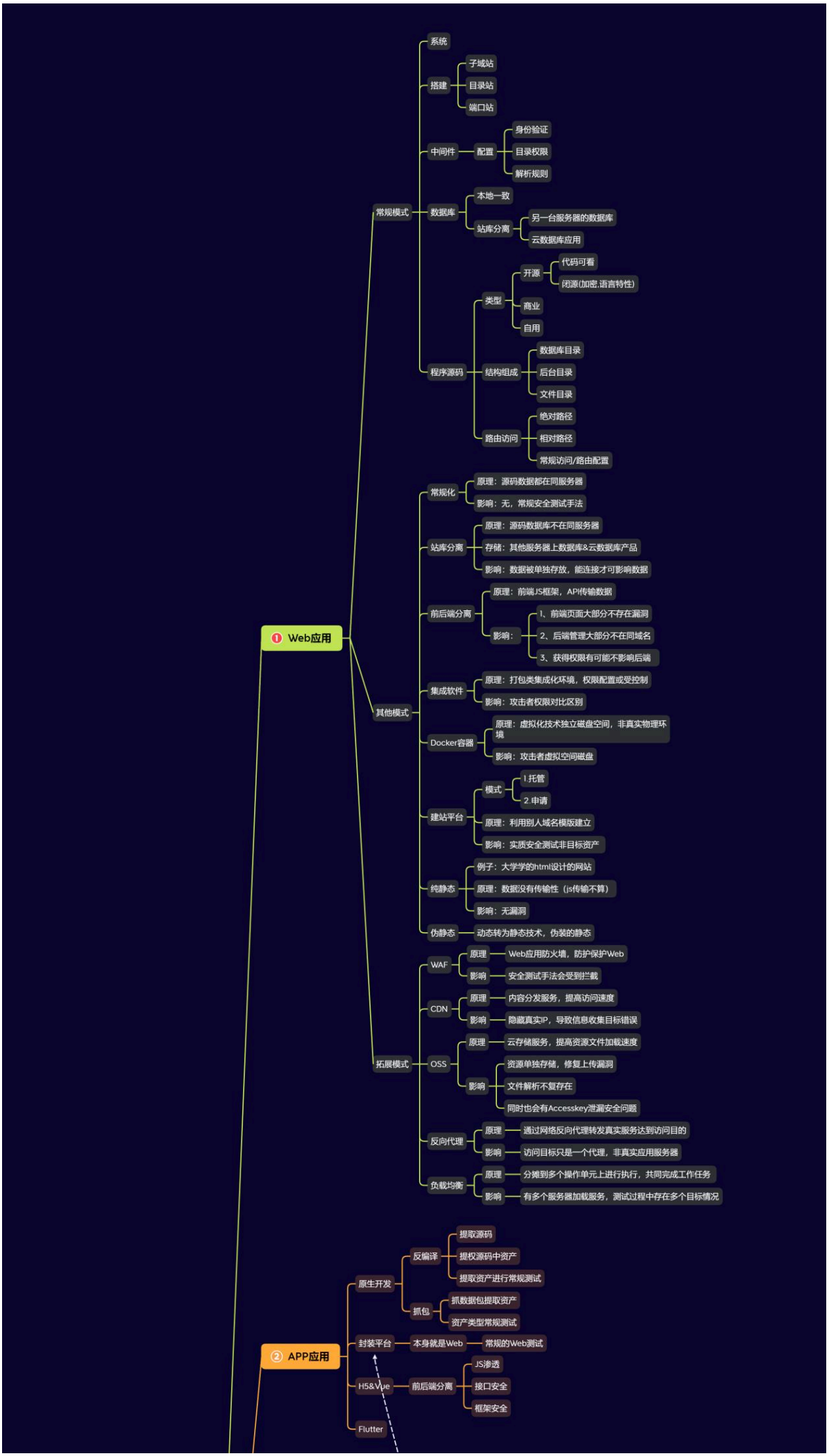
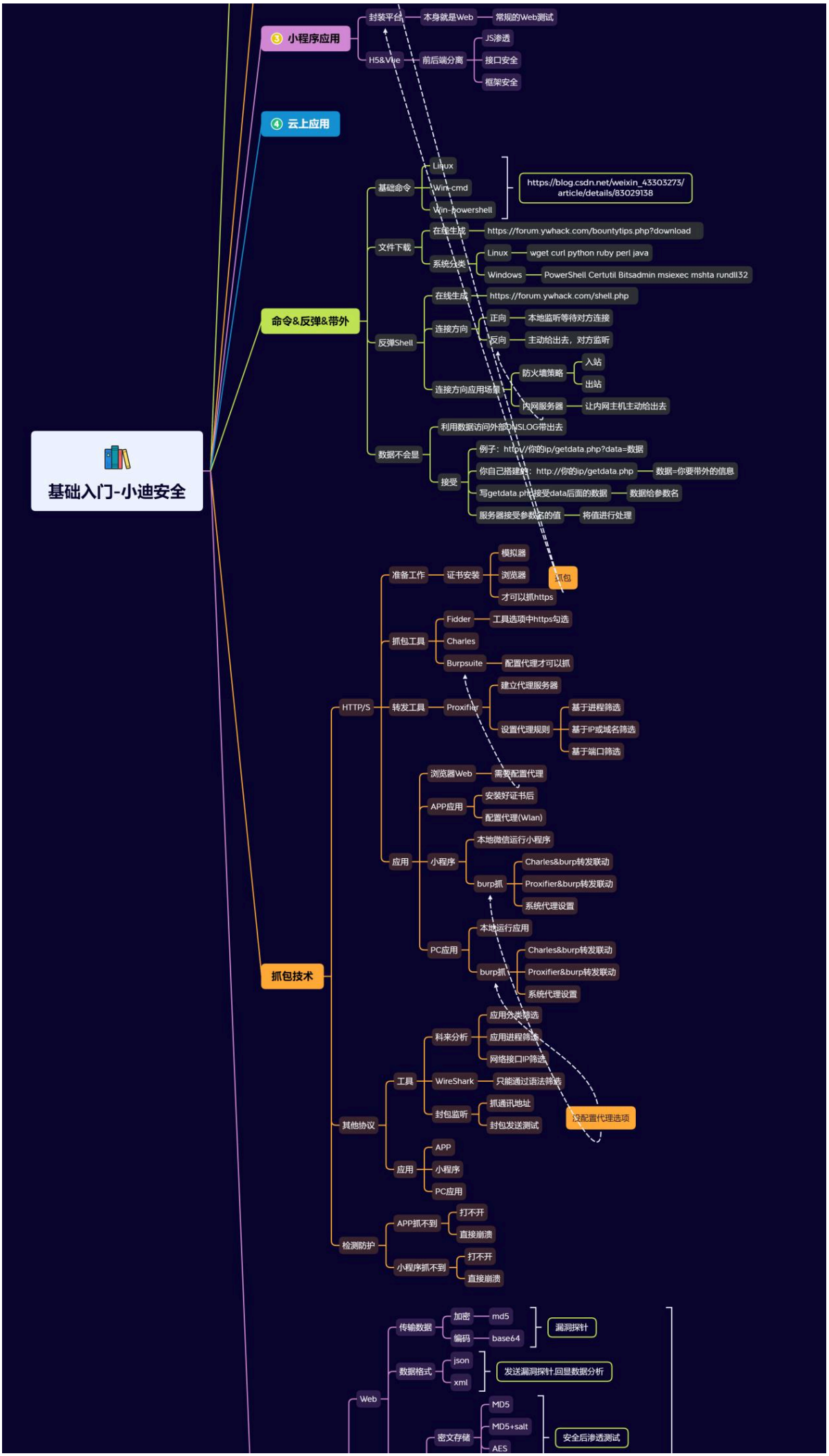
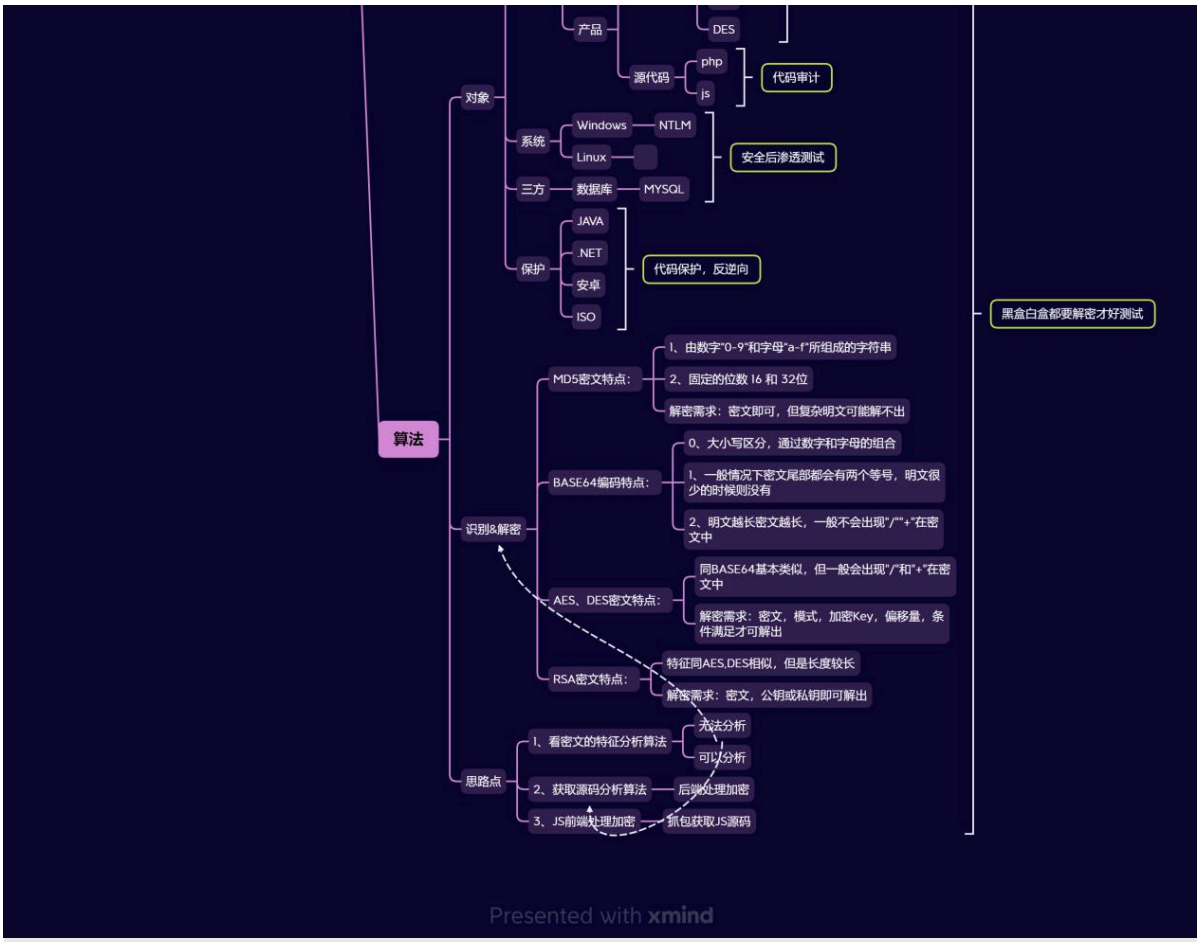
Day09　基础入门-算法逆向&散列对称非对称&JS源码逆向&AES&DES&RSA&SHA


① Web应用
常规模式
系统
搭建
子域站
目录站
端口站
中间件
配置
身份验证
目录权限
解析规则
数据库
本地一致
站库分离
另一台服务器的数据库
云数据库应用
程序源码
类型
开源
代码可看
闭源(加密,语言特性)
商业
自用
结构组成
数据库目录
后台目录
文件目录
路由访问
绝对路径
相对路径
常规访问/路由配置
其他模式
常规化
原理：源码数据都在同服务器
影响：无，常规安全测试手法
站库分离
原理：源码数据库不在同服务器
存储：其他服务器上数据库&云数据库产品
影响：数据被单独存放，能连接才可影响数据
前后端分离
原理：前端JS框架，API传输数据
影响：
1、前端页面大部分不存在漏洞
2、后端管理大部分不在同域名
3、获得权限有可能不影响后端
集成软件
原理：打包类集成化环境，权限配置或受控制
影响：攻击者权限对比区别
Docker容器
原理：虚拟化技术独立磁盘空间，非真实物理环境
影响：攻击者虚拟空间磁盘
建站平台
模式
1.托管
2.申请
原理：利用别人域名模版建立
影响：实质安全测试非目标资产
纯静态
例子：大学学的html设计的网站
原理：数据没有传输性（js传输不算）
影响：无漏洞
伪静态
动态转为静态技术，伪装的静态
拓展模式
WAF
原理
Web应用防火墙，防护保护Web
影响
安全测试手法会受到拦截
CDN
原理
内容分发服务，提高访问速度
影响
隐藏真实IP，导致信息收集目标错误
OSS
原理
云存储服务，提高资源文件加载速度
影响
资源单独存储，修复上传漏洞
文件解析不复存在
同时也会有Accesskey泄漏安全问题
反向代理
原理
通过网络反向代理转发真实服务达到访问目的
影响
访问目标只是一个代理，非真实应用服务器
负载均衡
原理
分摊到多个操作单元上进行执行，共同完成工作任务
影响
有多个服务器加载服务，测试过程中存在多个目标情况
② APP应用
原生开发
反编译
提取源码
提权源码中资产
提取资产进行常规测试
抓包
抓数据包提取资产
资产类型常规测试
封装平台
本身就是Web
常规的Web测试
H5&Vue
前后端分离
JS渗透
接口安全
框架安全
Flutter


基础入门-小迪安全
③ 小程序应用
封装平台
本身就是Web
常规的Web测试
H5&Vue
前后端分离
JS渗透
接口安全
框架安全
④ 云上应用
命令&反弹&带外
基础命令
Linux
Win-cmd
Win-powershell
https://blog.csdn.net/weixin_43303273/article/details/83029138
文件下载
在线生成
https://forum.ywhack.com/bountytips.php?download
系统分类
Linux
wget curl python ruby perl java
Windows
PowerShell Certutil Bitsadmin msiexec mshta rundll32
反弹Shell
在线生成
https://forum.ywhack.com/shell.php
连接方向
正向
本地监听等待对方连接
反向
主动给出去，对方监听
连接方向应用场景
防火墙策略
入站
出站
内网服务器
让内网主机主动给出去
数据不会显
利用数据访问外部DNSLOG带出去
接受
例子：http//你的ip/getdata.php?data=数据
你自己搭建的：http://你的ip/getdata.php
数据=你要带外的信息
写getdata.php接受data后面的数据
数据给参数名
服务器接受参数名的值
将值进行处理
抓包技术
HTTP/S
准备工作
证书安装
模拟器
浏览器
抓包
才可以抓https
抓包工具
Fidder
工具选项中https勾选
Charles
Burpsuite
配置代理才可以抓
转发工具
Proxifier
建立代理服务器
设置代理规则
基于进程筛选
基于IP或域名筛选
基于端口筛选
应用
浏览器Web
需要配置代理
APP应用
安装好证书后
配置代理(Wlan)
小程序
本地微信运行小程序
burp抓
Charles&burp转发联动
Proxifier&burp转发联动
系统代理设置
PC应用
本地运行应用
burp抓
Charles&burp转发联动
Proxifier&burp转发联动
系统代理设置
其他协议
工具
科来分析
应用分类筛选
应用进程筛选
网络接口IP筛选
WireShark
只能通过语法筛选
封包监听
抓通讯地址
封包发送测试
没配置代理选项
应用
APP
小程序
PC应用
检测防护
APP抓不到
打不开
直接崩溃
小程序抓不到
打不开
直接崩溃
Web
传输数据
加密
md5
编码
base64
漏洞探针
数据格式
json
xml
发送漏洞探针,回显数据分析
密文存储
MD5
MD5+salt
AES
安全后渗透测试



1.知识点

- 1、Web常规-系统&中间件&数据库&源码等
- 2、Web其他-前后端&软件&Docker&分配站等
- 3、Web拓展-CDN&WAF&OSS&反向&负载均衡等

- 1、APP架构-封装&原生态&H5&flutter等
- 2、小程序架构-Web&H5&JS&VUE框架等

- 1、渗透命令-常规命令&文件上传下载
- 2、反弹Shell-防火墙策略&正反向连接
- 3、数据回显-查询带外&网络协议层级

- 1、抓包技术-HTTP/S-Web&APP&小程序&PC应用等
- 2、抓包工具-Burp&Fidder&Charles&Proxifier

- 1、抓包技术-全局-APP&小程序&PC应用
- 2、抓包工具-Wireshark&科来分析&封包

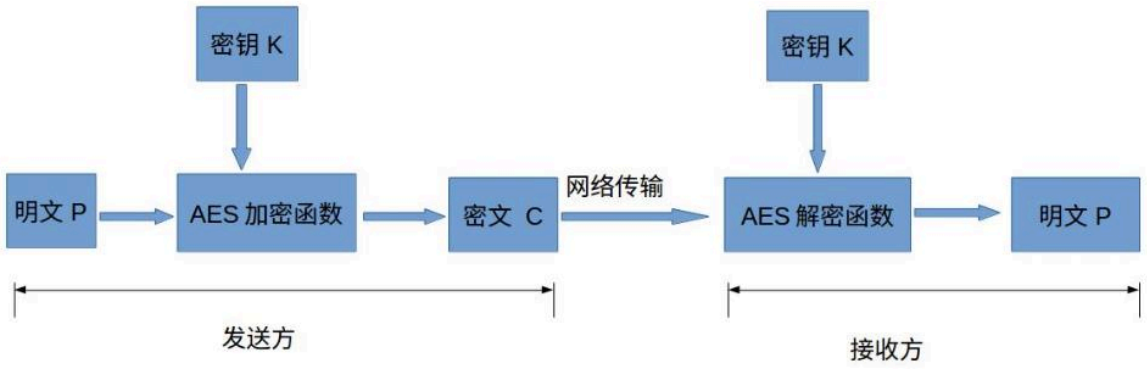
- 1、存储密码加密-应用对象
- 2、传输加密编码-发送回显
- 3、数据传输格式-统一格式
- 4、代码特性混淆-开发语言

- 1、单向散列加密 -MD5，HASH
- 2、对称加密 -AES DES
- 3、非对称加密 -RSA
- 4、解密-识别&需求&寻找(前后端)&操作

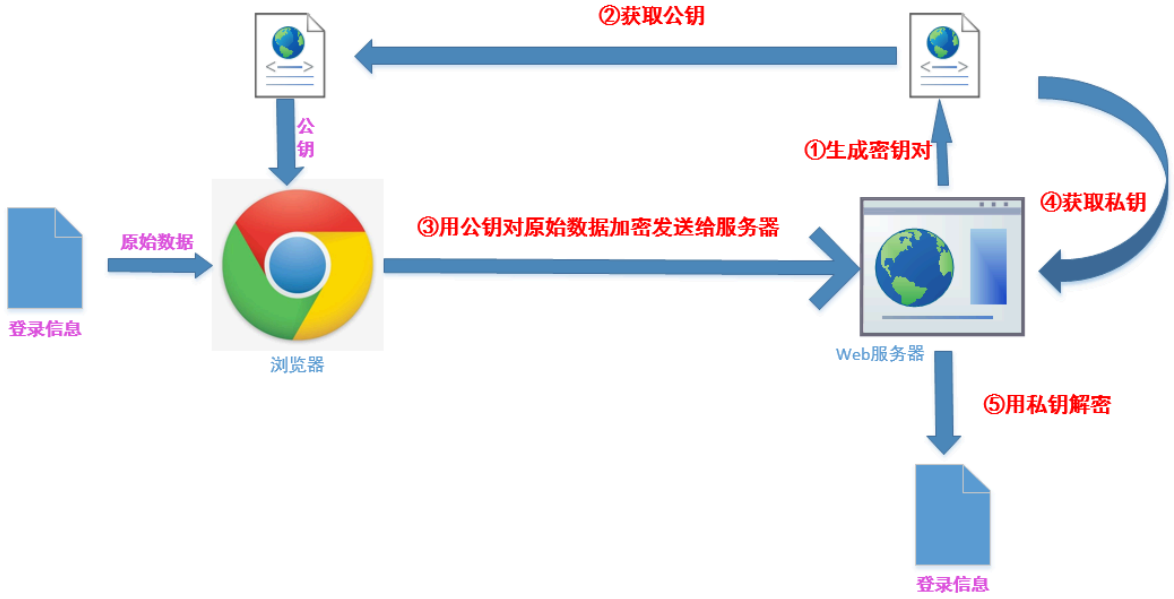
2.演示案例

```
安全测试中：
密文-有源码直接看源码分析算法（后端必须要有源码才能彻底知道）
密文-没有源码1、猜识别 2、看前端JS（加密逻辑是不是在前端）
```

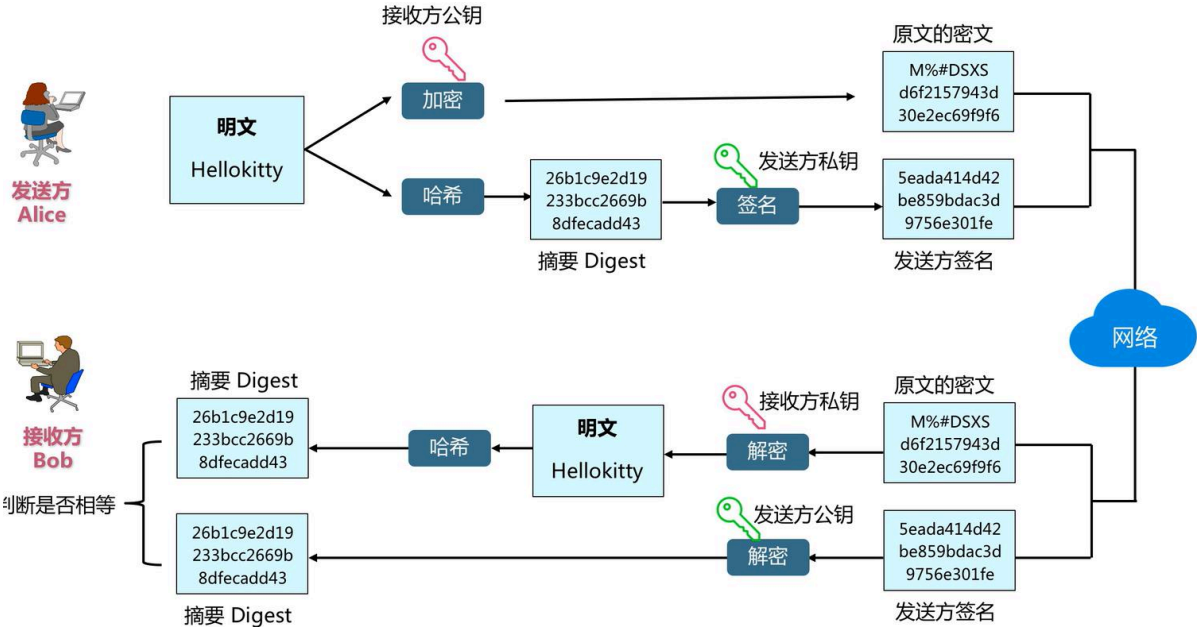
AES加密：



浏览器登录加解密过程：



非对称加密：



2.1 算法加密-概念&分类&类型

```
1. 单向散列加密 -MD5
单向散列加密算法的优点有(以MD5为例):
方便存储，损耗低：加密/加密对于性能的损耗微乎其微。
单向散列加密的缺点就是存在暴力破解的可能性，最好通过加盐值的方式提高安全性，此外可能存在散列冲突。我们都知道MD5加密也是可以破解的。
常见的单向散列加密算法有:
MD5 SHA MAC CRC

2. 对称加密 -AES
对称加密优点是算法公开、计算量小、加密速度快、加密效率高。
缺点是发送方和接收方必须商定好密钥，然后使双方都能保存好密钥，密钥管理成为双方的负担。
常见的对称加密算法有:
DES AES RC4

3. 非对称加密 -RSA
非对称加密的优点是与对称加密相比，安全性更好，加解密需要不同的密钥，公钥和私钥都可进行相互的加解密。
缺点是加密和解密花费时间长、速度慢，只适合对少量数据进行加密。
常见的非对称加密算法:
RSA RSA2 PKCS
```

2.2 加密解密-识别特征&解密条件

```
MD5密文特点:
1、由数字“0-9”和字母“a-f”所组成的字符串
2、固定的位数 16 和 32位
解密需求：密文即可，但复杂明文可能解不出
```

BASE64编码特点：
0、大小写区分，通过数字和字母的组合
1、一般情况下密文尾部都会有两个等号，明文很少的时候则没有
2、明文越长密文越长，一般不会出现"/""+"在密文中

AES、DES密文特点：
同BASE64基本类似，但一般会出现"/"和"+"在密文中
解密需求：密文，模式，加密Key，偏移量，条件满足才可解出

RSA密文特点：
特征同AES,DES相似，但是长度较长
解密需求：密文，公钥或私钥即可解出

其他密文特点见：
1.30余种加密编码类型的密文特征分析（建议收藏）
https://mp.weixin.qq.com/s?__biz=MzAwNDcxMjI2MA==&mid=2247484455&idx=1&sn=e1b4324ddcf7d6123be30d9a5613e17b&chksm=9b26f60cac517f1a920cf3b73b3212a645aeef78882c47957b9f3c2135cb7ce051c73fe77bb2&mpshare=1&scene=23&srcid=1111auAYWmr1NONAs9Wp2hGz&sharer_sharetime=1605145141579&sharer_shareid=5051b3eddbbe2cb698aedf94523700026#rd

2.CTF中常见密码题解密网站总结（建议收藏）
https://blog.csdn.net/qq_41638851/article/details/100526839

3.CTF密码学常见加密解密总结（建议收藏）
https://blog.csdn.net/qq_40837276/article/details/83080460

2.3 解密实例-密文存储&数据传输

```
1、密码存储（后端处理）
X3.2-md5&salt
DZ对应代码段-/uc_server/model/user.php
    function add_user() {
        $password = md5(md5($password).$salt);
    }

<?PHP
$h = 'd7192407bb4bfc83d28f374b6812fbcd';
$hash=md5(md5('123456').'3946d5');
if($h==$hash){
    echo 'ok';
}else{
    echo 'no';
}
?>

X3.5-hash
DZ对应代码段-/uc_server/model/user.php
    function add_user() {
        $salt = '';
        $password = $this->generate_password($password);
    }

    function generate_password($password) {
        $algo = $this->get_passwordalgo();
        $options = $this->get_passwordoptions();
        $hash = password_hash($password, $algo, $options);
```

```
    }

<?PHP
$hash =
'$2y$10$KA.7VYVheqod8F3X65tWjO3ZXfozNA2fC4oIZoDSu/TbfgKmiw7xO';
if (password_verify('123456', $hash)) {
    echo 'ok';
} else {
    echo 'error';
}
?>
```

```
2、数据通讯
-博客登录-zblog（前端处理）
<script src="script/md5.js"
type="text/javascript"></script>
$("#btnPost").click(function(){
    var strPassWord=$("#edtPassWord").val();
    $("form").attr("action","cmd.php?
act=verify");
    $("#password").val(MD5(strPassWord));

-墨者靶场-（后端处理）
-1 union select 1,database(),user(),4_mozhe
xgd58ipTrnx8VzSBJicqCibZxIRsZKgXOYUrNQP8fCCtx9JZ
+6K1hHt7RKkzV305
eGdkNThpcFRybng4VnpTQkppY3FDaWJaeElSc1pLZ1hPWVVy
TlFQOGZDQ3R4OUpaKzZLMWhIdDdSS2t6VjMwNQ==
```

```
//PHP7.3加密演示代码块
```

```php
<?php

//aes
namespace vendor;

class EncryptionTool{

    public static function enAES($originTxt, $key): string{

        return base64_encode(openssl_encrypt($originTxt, 'AES-128-ECB',$key, OPENSSL_RAW_DATA));
    }

    public static function deAES($originTxt, $key): string{

        $data = base64_decode($originTxt);
        return openssl_decrypt($data,'AES-128-ECB',$key, OPENSSL_RAW_DATA);
    }

}

//des
class DES
{
    /**
     * @var string $method 加解密方法，可通过 openssl_get_cipher_methods() 获得
     */
```

```
28      protected $method;
29      /**
30       * @var string $key 加解密的密钥
31       */
32      protected $key;
33      /**
34       * @var string $output 输出格式 无、base64、
    hex
35       */
36      protected $output;
37      /**
38       * @var string $iv 加解密的向量
39       */
40      protected $iv;
41      /**
42       * @var string $options
43       */
44      protected $options;
45      // output 的类型
46      const OUTPUT_NULL = '';
47      const OUTPUT_BASE64 = 'base64';
48      const OUTPUT_HEX = 'hex';
49      /**
50       * DES constructor.
51       * @param string $key
52       * @param string $method
53       *      ECB DES-ECB、DES-EDE3 （为 ECB 模式
    时，$iv 为空即可）
54       *      CBC DES-CBC、DES-EDE3-CBC、DESX-CBC
55       *      CFB DES-CFB8、DES-EDE3-CFB8
56       *      CTR
57       *      OFB
```

```
58          *
59          * @param string $output
60          *      base64、hex
61          *
62          * @param string $iv
63          * @param int $options
64          */
65         public function __construct($key, $method =
    'DES-ECB', $output = '', $iv = '', $options =
    OPENSSL_RAW_DATA | OPENSSL_NO_PADDING)
66         {
67             $this->key = $key;
68             $this->method = $method;
69             $this->output = $output;
70             $this->iv = $iv;
71             $this->options = $options;
72         }
73         /**
74          * 加密
75          *
76          * @param $str
77          * @return string
78          */
79         public function encrypt($str)
80         {
81             $str = $this->pkcsPadding($str, 8);
82             $sign = openssl_encrypt($str, $this-
    >method, $this->key, $this->options, $this-
    >iv);
83             if ($this->output ==
    self::OUTPUT_BASE64) {
84                 $sign = base64_encode($sign);
```

```
85          } else if ($this->output ==
    self::OUTPUT_HEX) {
86              $sign = bin2hex($sign);
87          }
88          return $sign;
89      }
90      /**
91       * 解密
92       *
93       * @param $encrypted
94       * @return string
95       */
96      public function decrypt($encrypted)
97      {
98          if ($this->output ==
    self::OUTPUT_BASE64) {
99              $encrypted =
    base64_decode($encrypted);
100         } else if ($this->output ==
    self::OUTPUT_HEX) {
101             $encrypted = hex2bin($encrypted);
102         }
103         $sign = @openssl_decrypt($encrypted,
    $this->method, $this->key, $this->options,
    $this->iv);
104         $sign = $this->unPkcsPadding($sign);
105         $sign = rtrim($sign);
106         return $sign;
107     }
108     /**
109      * 填充
110      *
```

```
111         * @param $str
112         * @param $blocksize
113         * @return string
114         */
115        private function pkcsPadding($str,
    $blocksize)
116        {
117            $pad = $blocksize - (strlen($str) %
    $blocksize);
118            return $str . str_repeat(chr($pad),
    $pad);
119        }
120        /**
121         * 去填充
122         *
123         * @param $str
124         * @return string
125         */
126        private function unPkcsPadding($str)
127        {
128            $pad = ord($str{strlen($str) - 1});
129            if ($pad > strlen($str)) {
130                return false;
131            }
132            return substr($str, 0, -1 * $pad);
133        }
134    }
135
136    //rsa
137    define('RSA_PUBLIC', '-----BEGIN PUBLIC KEY----
    -
```

```
138 MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCmkANmC84
    9IOntYQQdSgLvMMGm
139 8V/u838ATHaoZwvweoYyd+/7Wx+bx5bdktJb46YbqS1vz3V
    RdXsyJIWhpNcmtKhY
140 inwcl83aLtzJeKsznppqMyAIseaKIeAm6tT8uttNkr2zOym
    L/PbMpByTQeEFlyy1
141 poLBwrol0F4USc+owwIDAQAB
142 -----END PUBLIC KEY-----');
143
144 define('RSA_PRIVATE','-----BEGIN PRIVATE KEY---
    --
145 MIICdgIBADANBgkqhkiG9w0BAQEFAASCAmAwggJcAgEAAoG
    BAKaQA2YLzj0g6e1h
146 BB1KAu8wwabxX+7zfwBMdqhnC/B6hjJ37/tbH5vHlt2S0lv
    jphupLW/PdVF1ezIk
147 haGk1ya0qFiKfByXzdou3Ml4qzOemmozIAix5ooh4Cbq1Py
    62O2SvbM7KYv89syk
148 HJNB4QWXLLWmgsHCuiXQXhRJz6jDAgMBAAECgYAIF5cSriA
    m+CJlVgFNKvtZg5Tk
149 93UhttLEwPJC3D7IQCuk6A7Qt2yhtOCvgyKVNEotrdp3RCz
    ++CY0GXIkmE2bj7iO
150 fv5vT3kWvO9nImGhTBH6QlFDxc9+p3ukwsonnCshkSV9gmH
    5NB/yFoH1m8tck2Gm
151 BXDj+bBGUoKGWtQ7gQJBANR/jd5ZKf6unLsgpFUS/kNBgUa
    +EhVg2tfr9OMioWDv
152 MSqzG/sARQ2AbOO0ytpkbAKxxKkObPYsn47MWsf597OCQQD
    IqRiGmCY5QDAaejW4
153 HbOcsSovoxTqu1scGc3Qd6GYvLHujKDoubZdXCVOYQUMEnC
    D5j7kdNxPbVzdzXll
154 9+p/AkEAu/34iXwCbgEWQWp4V5dNAD0kXGxs3SLpmNpztLn
    /YR1bNvZry5wKew5h
```

```
z1zEFX+AGsYgQJu1g/goVJGvwnj/VQJAOe6f9xPsTTEb8jk
AU2S323BG1rQFsPNg
jY9hnWM8k2U/FbkiJ66eWPvmhWd7Vo3oUBxkYf7fMEtJuXu
+JdNarwJAAwJKOYmO
LxP4U+gTrj7y/j/feArDqBukSngcDFnAKu1hsc68FJ/vT5i
OC6S7YpRJkp8egj5o
pCcWaTO3GgC5Kg==
-----END PRIVATE KEY-----');



$password='xiaodisec';

//md5
echo "原始数据:$password". "<br/>";
echo "MD5加密后:".md5($password). "<hr/>";


//base64
echo "原始数据:$password". "<br/>";
echo "BASE64编码后:".base64_encode($password). "
<hr/>";


//aes
echo "原始数据: " . $password . "<br/>";
$data = EncryptionTool::enAES($password,
"1234567891234567");
echo "AES加密后: " . $data . "<hr/>";
//echo "解密后: " . EncryptionTool::deAES($data,
"1234567891234567") . "<br/>";
```

```php

//des
echo "原始数据:$password". "<br/>";
$key = 'key123456';
$iv = 'iv123456';
// DES CBC 加解密
echo 'DES CBC 加解密:';
$des = new DES($key, 'DES-CBC', DES::OUTPUT_BASE64, $iv);
echo $base64Sign = $des->encrypt($password);
echo "<br>";
//echo $des->decrypt($base64Sign);
echo "<hr>";
// DES ECB 加解密
echo "原始数据:$password". "<br/>";
echo 'DES ECB 加解密:';
$des = new DES($key, 'DES-ECB', DES::OUTPUT_BASE64);
echo $base64Sign = $des->encrypt($password);
echo "<hr>";
//echo $des->decrypt($base64Sign);


//rsa
//公钥加密
$public_key = openssl_pkey_get_public(RSA_PUBLIC);
if(!$public_key){
    die('公钥不可用');
}
```

```php
//第一个参数是待加密的数据只能是string，第二个参数是加密后的数据,第三个参数是openssl_pkey_get_public返回的资源类型,第四个参数是填充方式
$return_en = openssl_public_encrypt($password, $crypted, $public_key);
if(!$return_en){
    return('加密失败,请检查RSA秘钥');
}
$eb64_cry = base64_encode($crypted);
echo "RSA公钥加密数据: ".$eb64_cry;
echo "<br>";


//私钥解密
$private_key = openssl_pkey_get_private(RSA_PRIVATE);
if(!$private_key){
    die('私钥不可用');
}
$return_de = openssl_private_decrypt(base64_decode($eb64_cry), $decrypted, $private_key);
if(!$return_de){
    return('解密失败,请检查RSA秘钥');
}
echo "RSA私钥解密数据:".$decrypted;
echo "<hr>";


//私钥加密
$private_key = openssl_pkey_get_private(RSA_PRIVATE);
```

```php
if(!$private_key){
    die('私钥不可用');
}
$return_en = openssl_private_encrypt($password, $crypted, $private_key);
if(!$return_en){
    return('加密失败,请检查RSA秘钥');
}
$eb64_cry = base64_encode($crypted);
echo "RSA私钥加密数据".$eb64_cry;
echo "<br>";

//公钥解密
$public_key = openssl_pkey_get_public(RSA_PUBLIC);
if(!$public_key){
    die('公钥不可用');
}
$return_de = openssl_public_decrypt(base64_decode($eb64_cry), $decrypted, $public_key);
if(!$return_de){
    return('解密失败,请检查RSA秘钥');
}
echo "RSA公钥解密数据:".$decrypted;
echo "<hr>";

?>
```

资源：

```
1.30余种加密编码类型的密文特征分析（建议收藏）
https://mp.weixin.qq.com/s?__biz=MzAwNDcxMjI2MA==&mid=2247484455&idx=1&sn=e1b4324ddcf7d6123be30d9a5613e17b&chksm=9b26f60cac517f1a920cf3b73b3212a645aeef78882c47957b9f3c2135cb7ce051c73fe77bb2&mpshare=1&scene=23&srcid=1111auAYwmr1NONAs9Wp2hGz&sharer_sharetime=1605145141579&sharer_shareid=5051b3eddbbe2cb698aedf9452370026#rd
2.CTF中常见密码题解密网站总结（建议收藏）
https://blog.csdn.net/qq_41638851/article/details/100526839
3.CTF密码学常见加密解密总结（建议收藏）
https://blog.csdn.net/qq_40837276/article/details/83080460
```