

加密分析（逆向）

陈梓轩

对中大网校登录的安全认证、加密过程分析

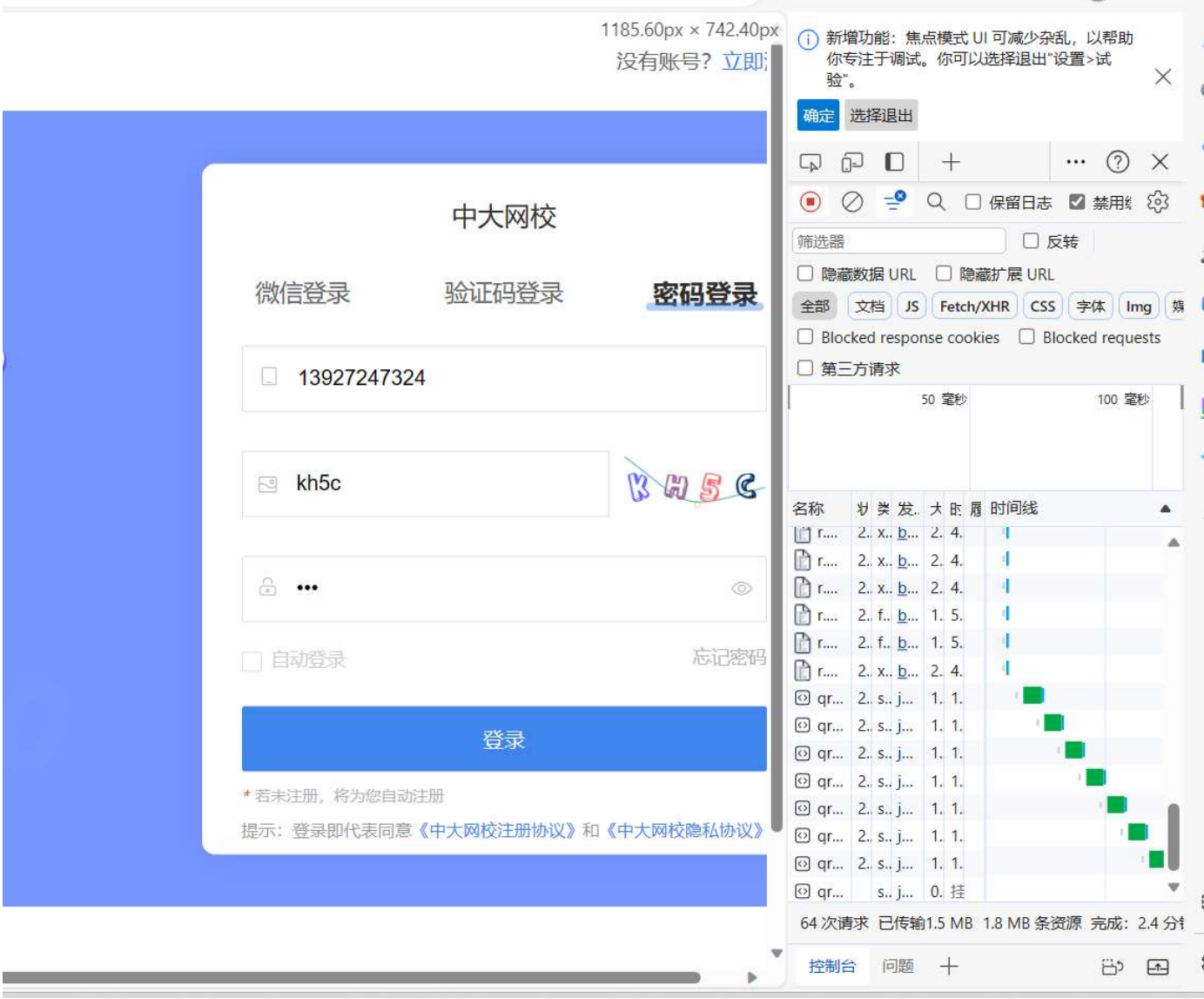
中大网校登录url: <https://user.wangxiao.cn/login> (<https://user.wangxiao.cn/login>)

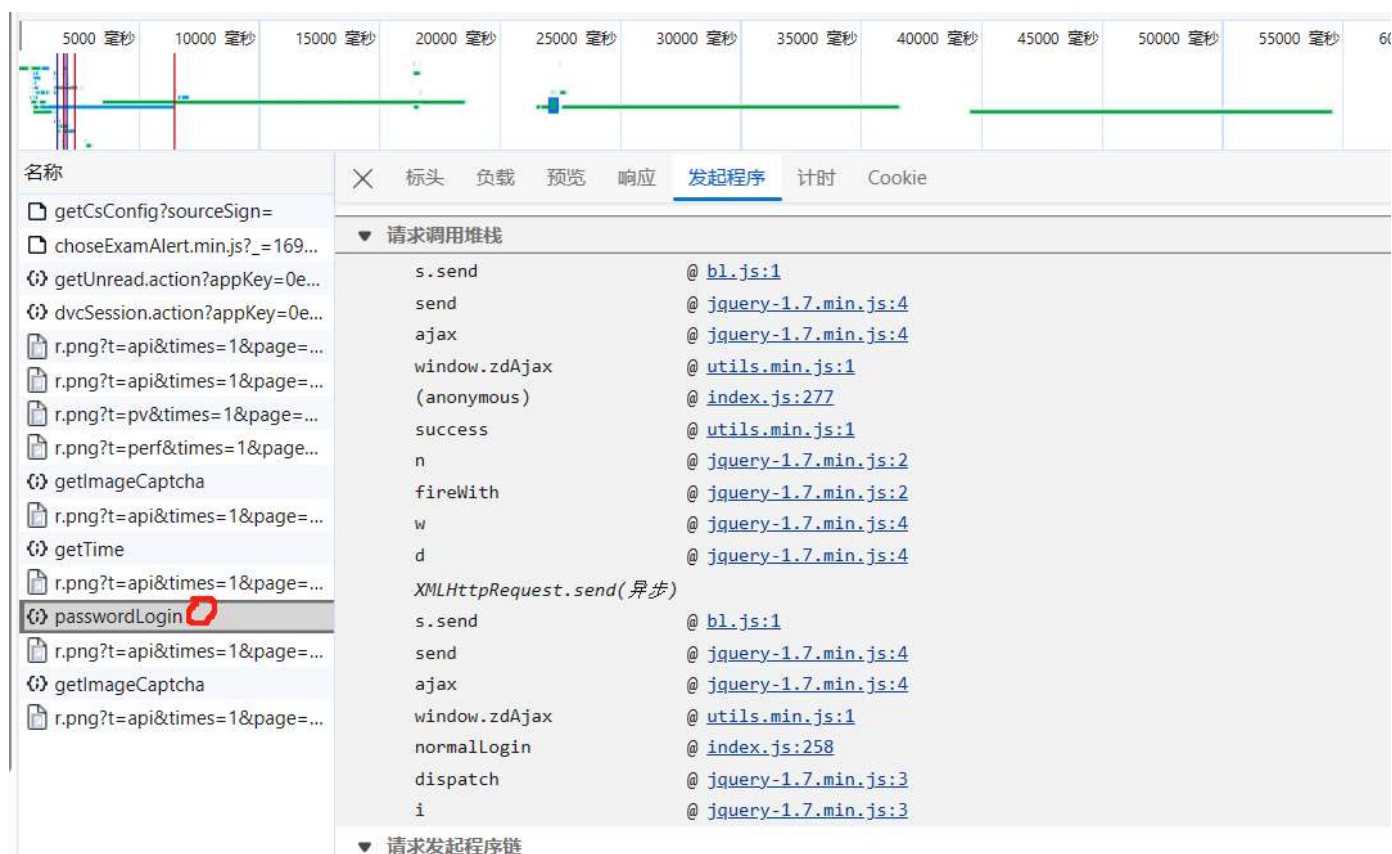
拟利用F12，并进行登录，抓取在登录过程中传输的包，设置断点并反复提交登录，确定前端加密程序的具体位置，对其进行逆向工程完成加密分析

实验流程

登录界面抓包

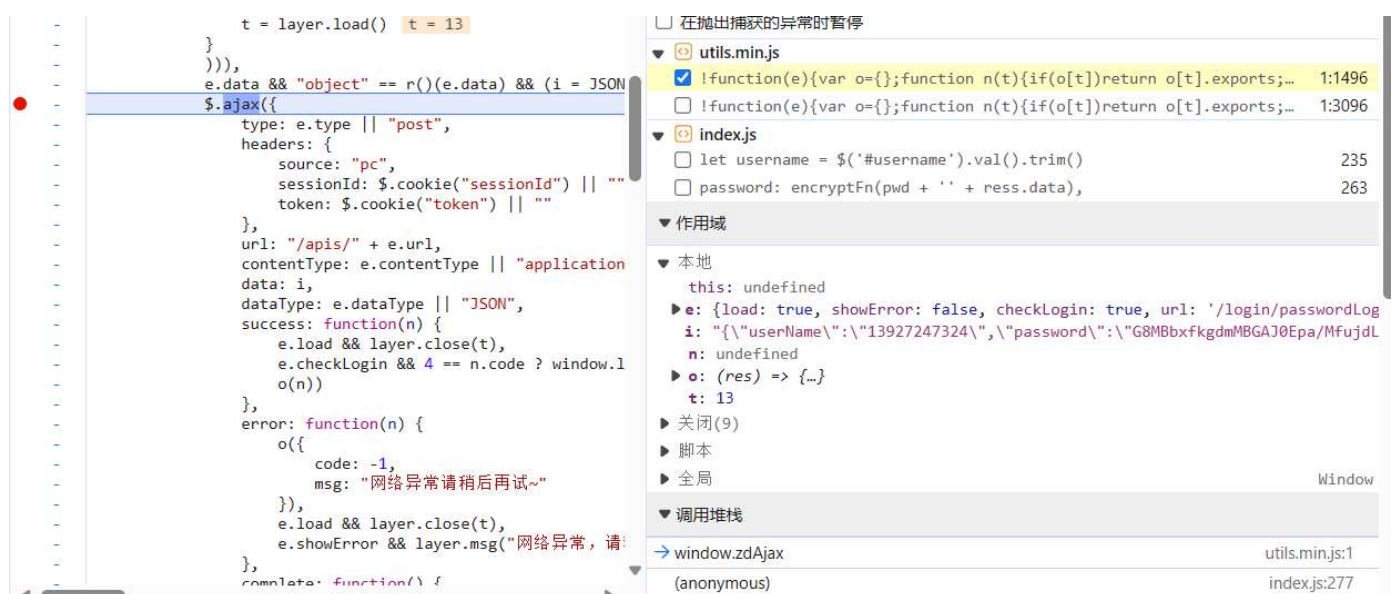
对中大网校登录页面F12抓包





观察捕获的包，其中一包ID名为passwordLogin，猜测大概率为所需包，在passwordLogin包中涉及许多程序，从上往下进行分析，容易知道jquery.js文件不为所需，尝试bl.js:1文件无所需加密信息，查看utils.min/js:1文件（堆栈window.zdAjax），通过后续过程发现加密程序确实该文件中，进行逆向。

设置断点定位加密位置



通过设置断点并观察password是否是已加密的状态，层层向上寻找，找到令password加密的位置，定位加密程序

```
var t = n(2)
, r = n.n(t);
window.zdAjax = function(e, o, n) { e = {url: '/login
var t, i; t = undefined, i = undefined
e = Object.assign({
  load: !0,
  showError: !1,
  checkLogin: !0
}, e)).load && (t = layer.load(),
layer.ready(function(e) {
  layer.close("loading"),
  t = layer.load()
}
)),
e.data && "object" == r()(e.data) && (i = JSON.str
$.ajax({
  type: e.type || "post",
  headers: {
    source: "pc",
    sessionId: $.cookie("sessionId") || "",
    token: $.cookie("token") || ""
  },
  url: "/apis/" + e.url,
  contentType: e.contentType || "application/jsor
  data: i,
  dataType: e.dataType || "JSON",
  success: function(n) {
    e.load && layer.close(t),
```

在抛出捕获的异常时暂停

utils.min.js

☒

!function(e){var o={};function n(t){if(o[t])return o[t].exports;var r=o...

1:1289

☐

!function(e){var o={};function n(t){if(o[t])return o[t].exports;var r=o...

1:13096

index.js

☐

let username = \$('#username').val().trim()

235

☐

password: encryptFn(pwd + '' + ress.data),

263

作用域

本地

this: undefined

e:

data: {userName: '13927247324', password: 'p7nYRtqrTC53Ah/vk12MjIQsiyYGNVusEltLc...

url: "/login/passwordLogin"

[[Prototype]]: Object

constructor: f Object()

hasOwnProperty: f hasOwnProperty()

isPrototypeOf: f isPrototypeOf()

propertyIsEnumerable: f propertyIsEnumerable()

toLocaleString: f toLocaleString()

toString: f toString()

valueOf: f valueOf()

__defineGetter__: f __defineGetter__()

__defineSetter__: f __defineSetter__()

__lookupGetter__: f __lookupGetter__()

__lookupSetter__: f __lookupSetter__()

```
// } ;
//去请求接口
zdAjax(param, (res) => {
  if (res.code == 0) {
    if ($('#auto-login').is(':checked')) {
      //自动登录
      keepOurCookie12('autoLogin', true, :
      keepOurCookie12('userInfo', JSON.str
      keepOurCookie12('token', res.data.t
      syncLogin(res.data, expiresDay)
    } else {
      keepOurCookie12('autoLogin', null)
      keepOurCookie12('userInfo', JSON.str
      keepOurCookie12('token', res.data.t
      syncLogin(res.data)
    }
  }
  login.jump(res.data.isBindingMobile)
} else if (res.code == '9') {
  //密码错误达到了两次
  login.getImgCode($('#nimg-code .img-c
  $('#nimg-code').addClass('show')
```

在抛出捕获的异常时暂停

utils.min.js

☐

let username = \$('#username').val().trim()

235

☐

password: encryptFn(pwd + '' + ress.data),

263

☒

zdAjax(param, (res) => {

277

作用域

本地

this: undefined

param:

data: {userName: '13927247324', password: 'OK/EgjWlCTha1bK6Q/Q2Q+jtIUXW8v5vZpS4c...

url: "/login/passwordLogin"

[[Prototype]]: Object

ress: {code: 0, msg: null, data: '1699275000542', operation_date: '2023-11-06 20:5

关闭(normalLogin)

脚本

全局

调用堆栈

→ (anonymous)

index.js:277

success

utils.min.js:1

最终找到加密程序位置
由程序可知该加密过程：
对password+ress.data一同进行encrypt（加密程序）
其中ress为 url: '/common/getTime' 的json响应

```
var param = { param: {url: '/login/passwordLogin', data: {
  url: '/login/passwordLogin',
  data: {
    userName: username,
    password: encryptFn(pwd + '' + ress.data),
    imageCaptchaCode: imgCode,
  },
  // let param = {
  //   url: '/login/passwordLogin',
  //   data: {
  //     userName: username,
  //     password: encryptFn(pwd),
  //     imageCaptchaCode: imgCode
  //   }
  // };
  //去请求接口
zdAjax(param, (res) => {
  if (res.code == 0) {
    if ($('#auto-login').is(':checked')) {
      //自动登录
      keepOurCookie12('autoLogin', true, 30)
      keepOurCookie12('userInfo', JSON.stringify(res.data), expiresDay)
      keepOurCookie12('token', res.data.token, expiresDay)
      syncLogin(res.data, expiresDay)
    } else {
      keepOurCookie12('autoLogin', null)
      keepOurCookie12('userInfo', JSON.stringify(res.data))
```

在抛出捕获的异常时暂停

utils.min.js

☐

!function(e){var o={};function n(t){if(o[t])return o[t].exports...

1

☒

let username = \$('#username').val().trim()

235

☒

password: encryptFn(pwd + '' + ress.data),

263

作用域

本地

this: undefined

param:

data: {userName: '13927247324', password: 'Nji0Zk9KSb5wLWoTs0JDiiIetNS

url: "/login/passwordLogin"

[[Prototype]]: Object

ress: {code: 0, data: "1699275048085", msg: null, operation_date: "2023-11-06 20:50:48"

[[Prototype]]: Object

关闭(normalLogin)

脚本

全局

调用堆栈

→ (anonymous)

index.js:277

进入加密程序encrypt

加密程序encryptFn如图：

```
window.encryptFn = function(e) {  
    var o = new JSEncrypt;  
    return o.setPublicKey("MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDA5Zq6ZdH/RMSvC8WKhp5gj6  
    o.encrypt(e)  
}
```

根据该程序，并进入JSEncrypt查看，可以知道中大网校登录利用了RSA公钥加密算法对登录过程的密码（准确地说，被加密的不只是密钥，还有gettime.data）在传输前进行了加密，并且该程序中我们可以看到给定的公钥，所以后续我们可以利用该公钥并通过requests包输入指定data，登陆中大网校。

实现登录程序

完成登录过程需要四步

1. 进入登录页面
2. 验证码
3. 对密码进行加密
4. 进行登录

进入登录页面

```
url = "https://user.wangxiao.cn/login"  
sess = requests.session()  
resp = sess.get(url)
```

验证码

我们可以从登录页面的cookies中获取图片，利用图像处理api获取验证码。
在抓包过程可知验证码图片来源于/getImageCaptcha页面，并可在包中获取其url。

```

verify_code_url = "https://user.wangxiao.cn/apis//common/getImageCaptcha"
verify_resp = sess.post(verify_code_url,headers = headers)
verify_dic = verify_resp.json()
verify_b64_img = verify_dic["data"].split(",")[-1]

# 图像处理利用图片获取验证码程序
def base64_api(img):
    data = {"username":username,"password":password,"typeid"=typeid,"image":img}
    result = json.loads(
        requests.post("http://api.ttshitu.com/predict",json=data).text
    )
    if resule['success']:
        return result["data"]["result"]
    else:
        return result["message"]

verify_code = base64_api(verify_b64_img)

```

对密码进行加密

通过逆向工程，利用源程序中提供的公钥和RSA公钥加密算法，对密码进行加密。

```

gettime_url = "https://user.wangxiao.cn/apis//common/getTime"
gettime_resp = sess.post(gettime_url,headers = headers)
gettime_data = gettime_resp.json()["data"]

public_key = "-----BEGIN RSA PUBLIC KEY-----
\nZ0Cg7opDNYJM/lFDbRIILRx3pp0HMEkY5Cc/39s8CxDsNVeMw3A+TYi+4bsHyJajR9st/6ib4
KGa9+rCS98JM/z8oOLIMAMQVCJ2ehV2fr17SjLdUH3pDlN1A3idTK7crUdd/UUUhJee9pDTI6LI
00X7wH0hBpMb85TUs1aDteE=\n-----END PUBLIC KEY-----"

key = RSA.import_key(public_key)
rsa = PKCS1_v1_5.new(key)

# 加密    encryptFn(pwd + '' + ress.data)

password = password +str(gettime_data)
miwen = rsa.encrypt(password.encode("utf-8"))
miwen = base64.b64encode(miwen).decode("utf-8")

```

进行登录

获取登录页面的响应，并提供所需的数据（用户名，密码，验证码），完成登录


```
login_url = "https://user.wangxiao.cn/passwordLogin"
login_data = {
    "imageCaptchaCode": verify_code
    "password":miwen
    "userName":username
}

login_resp = sess.post(login_url, data = json.dumps(login_data), headers = headers)
print(login_resp.text)
```

- 可以见到返回成功的响应:

```
{"code":0,"msg":"成功","data":{"userName":"pc_783235612","token":"60cb3570-8138-4706-9d95-95be1784f861","headImg":null,"nickName":"186***5987",
"sign":"fangchan","isBindingMobile":"1","isSubPa":"0","userNameCookies":"x0WDyiWtCnEUVEjvM6TyDA==",
"passwordCookies":"NuP+k21mg/tu3w1FHGZ6Vg=="}, "operation_date":"2021-09-09 22:01:52"}
```

结论

在抓包及实验分析的过程中，我们知道了中大网校登录的加密程序，是无js混淆的RSA公钥加密算法。

中大网校登录过程中，会在前端对输入的明文密码结合gettime界面上的参数，转换为utf-8编码，并利用给定的公钥进行加密，最后利用base64将字节转换为字符串，并进行utf-8解码然后进行传输。

而在登录验证过程中，会将传输得到的密码密文与数据库中的密码密文进行比对（存储在数据库中的也是密文以保证密码隐私安全）

利用RSA非对称加密算法，并且在加密时加入了对原有密码明文添加了时间参数，RSA基于大素数分解难问题，在前端对密码进行加密，可以很好地保证传输和存储时密码的安全性。