[XCTF-Reverse] 86 XCTF 3rd-GCTF-2017_密码加密

原创

石氏是时试 于 2022-03-29 22:51:15 发布 41 收藏
分类专栏： CTF reverse 文章标签： reverse

一个题怎么运行的不大清楚，不过用处也不大，知道是个jar就行了

是个网页pwdmod.jsp的action是dopwdset.jsp

dopwdset.jsp 将数据发送给com.zte.zxywpub.Purview->changePwd(String,String)->changePwd(String,String,String,boolean)

在这里给输入的密码加个"11"然后生成对象encrypt（在res.jar里）这个对象继承BaseEncrypt,在这里调用方法VarLenEncrypt在这里会选择加密方式。

```
public void changePwd(String operid, String newpwd, String param, boolean isDefault) throws Exception {
  try {
    PwdRule rule = new PwdRule();
    HashMap map = getOperInformation(operid);
    String operName = (String)map.get("OPERNAME");
    String histPwd = rule.getHistoryPwd(operid, isDefault);
    if (histPwd.indexOf("\"" + newpwd + "\"") >= 0)
      throw new Exception("password already used.");
    new encrypt();
    String encryptepwd = encrypt.VarLenEncrypt(clearPwd(newpwd) + "11", 30);
```

通过config/app/encflag的值来确定加密方式，但这里没设置，所以走到

```
classicVarLenEncrypt(str_crypt_in, str_num);
```

```
public static String VarLenEncrypt(String str_crypt_in, int str_num) {
  XmlConfig config = XmlConfig.getInstance("com/zte/config.xml");
  ConfigNode node = config.selectSingleNode("/config/app/encflag");
  String encflag = ConfigTypeUtil.toString((node == null) ? null : node.getText(), "0");
  if (str_num == -1) {
    if ("1".equals(encflag)) {
      AES aes = new AES();
      String result = aes.VarEncryptPassword(str_crypt_in, str_num);
      return encryptPwSplit(result);
    }
    if ("2".equals(encflag) && str_crypt_in.matches("^!@#\\$%\\^.*$")) {
      AES aes = new AES();
      String result = aes.VarEncryptPassword(str_crypt_in.substring(6), str_num);
      return encryptPwSplit(result);
    }
    return classicVarLenEncrypt(str_crypt_in, str_num);
  }
  if (str_num >= 18 && str_num >= str_crypt_in.length()) {
    if ("1".equals(encflag)) {
      AES aes = new AES();
      String result = aes.VarEncryptPassword(str_crypt_in, str_num);
      return encryptPwSplit(result);
    }
    if ("2".equals(encflag) && !str_crypt_in.matches("^[a-z[A-Z[_[0-9]]]]*$")) {
      AES aes = new AES();
      String result = aes.VarEncryptPassword(str_crypt_in, str_num);
      return "!@#$%^" + encryptPwSplit(result);
    }
    return classicVarLenEncrypt(str_crypt_in, str_num);  //<----- 入口
  }
  return "";
}
```

这个函数写得很好，没有任何交叉引用。所以就可以直接把这个函数拿出来稍加修改运行就成了。

只是函数很长而且有乱字符（不能ctrl-c），需要用jd-jui另外源码然后再复制。

```
import java.io.*;

public class xxx
{
          public static void main(String[] args) {
            int KEY_LEN = 18, MAX_STR_LEN = 50;
/*   42 */     String str_dest = "";
/*   43 */     char[] str_crypt = "admin12345611".toCharArray();
/*      */     int str_num = 30;
/*      */
/*   46 */     int[] lastnum = new int[8];
/*      */
/*   48 */     char[] str_result = new char[MAX_STR_LEN];
/*   49 */     char[] str_result1 = new char[MAX_STR_LEN];
/*   50 */     char[] longkey = new char[MAX_STR_LEN];
/*   51 */     char[] str_crypt1 = new char[1];
/*   52 */     char[] str_key = new char[MAX_STR_LEN];
/*   53 */     char[][] key = new char[8][KEY_LEN];
.......中间略掉，修改很小，只删了几个this。
/*      */     System.out.println(str_dest);
/* 1008 */     return;
/*      */   }
/*      */ }
```

```
D:\xctf.rev\a86_XCTF 3rd-GCTF-2017_密码加密>javac xxx.java

D:\xctf.rev\a86_XCTF 3rd-GCTF-2017_密码加密>java xxx
1duGSe3JXvyMGYbpryYSeknbsWYWyU
```