

# 01-Atlassian明文密码登录

## 问题

在应用表单登录的时候，我们填写用户名和密码，点击登录提交后，提交的信息是以明文向后台发送请求



欢迎访问 汇科天下

用户名

密码

☒ 记住我在此计算机的登录

还不是会员? 可以请求帐户请联系 Jira 管理员。

[登录](#) [无法访问帐户?](#)



POST http://jira.jiracn.com/login.jsp

消息头 Cookie 参数 响应 耗时

过滤请求参数

表单数据

- os\_username: "admin"
- os\_password: "admin"
- os\_cookie: "true"
- os\_destination: ""
- user\_role: ""
- atl\_token: ""
- login: "登录"

这样，很容易被非法的人通过此漏洞获得个人的账户和密码信息，给系统安全带来很大的风险。

## 解决方案

用户在登录表单中填写正常的账户名和密码，进行提交；

在提交的动作插入加密码后的账户和密码，然后提交加密后的账户和密码到后台；

后台接收到登录请求，对账户和密码进行解密，并以用户真正的账户和密码进行身份验证

如可以优化，可以在后台生成临时的登录校验码，并加入到加密的运算中，然后在后台获得提交的信息，加入保存的过校验码进行解析（校验码只能使用一次）

## 解决方法

### 前端（JIRA）

修改/includes/loginform.jsp

## 前端 (Confluence)

修改/login.vm

### 修改示例

控制提交按钮的行为, 可在页面底引入js, 通过js进行加密;

```
<SCRIPT LANGUAGE="JavaScript">
<!--
function encrypt(value) {
    return "加密后的值"
}
function submitLogin() {
    var pwd = jQuery("#login-form-password").val();
    var user = jQuery("#login-form-username").val();
    var encryptPass = encrypt(pwd);
    var encryptUser = encrypt(user);
    jQuery("#login-form-password").val(encryptPass);
    jQuery("#login-form-username").val(encryptUser);
    jQuery("#login-form").submit();
}
//-->
</SCRIPT>
```

## 后端

后台对request获得的账户名和密码进行加密, 并获得解密后的账户名和密码进行后续的验证。

注意验证包含以下几种场景

- 本地验证 (jira本地目录的用户)
- LDAP验证 (交由LDAP进行验证)
- Authorization 的认证方式

可修改以下类中的方法:

- com.atlassian.seraph.filter.LoginFilter#extractUserPasswordPair
- com.atlassian.seraph.filter.HttpAuthFilter#extractUserPasswordPair

### LoginFilter

```

protected UserPasswordPair extractUserPasswordPair (HttpServletRequest request)
{
    // check for parameters
    String username = request.getParameter (RequestParameterConstants.
OS_USERNAME);
    String password = request.getParameter (RequestParameterConstants.
OS_PASSWORD);
    String newusername = "解密后的用户名";
    String newpassword = "解密后的密码";

    boolean persistentLogin = "true".equals (request.getParameter
(RequestParameterConstants. OS_COOKIE));
    return new UserPasswordPair (newusername, newpassword, persistentLogin);
}
}

```

#### HttpAuthFilter

```

protected UserPasswordPair extractUserPasswordPair (HttpServletRequest request)
{
    String auth = request.getHeader ("Authorization");
    if (SecurityUtils.isBasicAuthorizationHeader (auth))
    {
        SecurityUtils.UserPassCredentials creds = SecurityUtils.
decodeBasicAuthorizationCredentials (auth);
        if (!"".equals (creds.getUsername ()))
        {
            String username = creds.getUsername ();
            String password = creds.getPassword ();

            String newusername = "解密后的用户名";
            String newpassword = "解密后的密码";

            return new UserPasswordPair (newusername, newpassword, false);
        }
    }
    return null;
}

```

汇科天下版权所有，请勿转载

Anonymous(3.144.42.196)

2024-04-19 02:04:24

汇科天下版权所有，请勿转载

Anonymous(3.144.42.196)

2024-04-19 02:04:24

汇科天下版权所有，请勿转载

Anonymous(3.144.42.196)

2024-04-19 02:04:24

汇科天下版权所有，请勿转载

Anonymous(3.144.42.196)

2024-04-19 02:04:24

汇科天下版权所有，请勿转载

Anonymous(3.144.42.196)

04:24

汇科天下版权所有，请勿转载

Anonymous(3.144.42.196)

04:24