Thunderhead Experience

I look forward to getting to use Thunderhead now that I have things working. The following document highlights some of the challenges I faced and about how much time I spent on them to help point out challenge points. This document is not meant to be overly critical, but merely to help improve the user experience with Thunderhead.

I. GSDK and Thunderhead Specific Challenges

1. Windows Version
    a. In the Locally debugging game servers and integration with PlayFab tutorial and **Verifying Containerization** section, the prerequisites say "Windows 10 with April 2018 (1803) update." It is missing one prerequisite which is "Windows Pro or above". Windows Home is not capable of using Hyper-V, and therefore can't use Docker. That would have saved me about 1 day of debugging.
2. Firewall
    a. The description of *Setup.ps1* suggests that running that will set everything up in the firewall. However, it does not account for 3rd party antiviruses such as McAfee (in my case). If there was a disclaimer like "you may have to configure / turn off the firewall of your 3rd party antivirus such as McAfee, Norton, Avira, etc." it would have saved me about 1 day of debugging.
3. GSDK samples from github
    a. When using this sample and going through the Locally debugging game servers and integration with PlayFab tutorial, the sample will not work with the **Verifying GSDK integration.** The user will get authorization errors unless they run the .exe in Administrator mode. If the user is like me, they will then think "I can't run the .exe in Administrator mode in the container. Let me find the problem and fix it." Without realizing they don't need to run the program in Administrator mode in the container because it is in the container. Anyways, I changed the code from

```
string address = $"http://*:{listeningPort}/";
```

to

```
string address = $"http://localhost:{listeningPort}/";
```

That makes it work for the **Verifying GSDK integration** part without needing to run in administrator mode, but breaks in the **Verifying Containerization** part of the tutorial. If there were directions in the **Verifying GSDK integration** part that said "If using the GSDK samples from github, run the .exe in in Admistrator mode." That would have saved me about 3 days of debugging by preventing me from going down the rabbit hole of changing that line of code above.

1.) RequestMultiplayerServer API Call

a. Everything about this API call was straight forward today except for the **SessionId.** The docs state, "A guid string session ID created track the multiplayer server session over its life." I kept looking for the SessionID in the GameManager to provide but couldn't find it. I then tried creating the GUID myself because I couldn't think of anything else to do and it worked. If the description was changed to something like, "A guid string **created by the caller** to track the multiplayer server session over its life." I would have understood it. This would have saved me about 1 hour of debugging.

II. Unity and Thunderhead Challenges

I realize there isn't any official Unity support yet, but there are some things that can be done to point us Unity developers in the right direction.

1.) Game Server SDK nuget package
   a. This part only took me about 15 minutes to solve, but I imagine could be a struggle for other Unity developers. Unity doesn't have official NuGet support, but they do support placing external dll's in the Assets/Plugins folder. All you have to do as a unity developer is download the Game Server SDK nuget package, extract its contents with something like 7-Zip, and place the Microsoft.Playfab.Gaming.GSDK.CSharp.dll in the Plugins folder as well as make sure Json.NET is in the project. I personally just use NuGet For Unity to get Json.Net. NuGet For Unity **DOES NOT** work for the Game Server SDK nuget package. That is why you have to download, extract it, and place it in the Plugins folder. I believe you can add Unity support to a NuGet package as described here, but I haven't tested it out.
2.) UNet uses UDP
   a. This part was my fault. I'm not sure why, but I though UNet listens on TCP. If I would have simply changed the protocol to UDP in the MultiplayerSettings.json file, everything would have been working 5 days ago. If you could have a docs section for UNet that states some of these things, it could help other Unity developers. I know this part is small and was my responsibility to know, but if it was stated somewhere it would have saved me about 5 days of debugging.

III. Conclusion

Overall, I really look forward to using Thunderhead and find the local debugger extremely useful. The local debugger has saved me hours, maybe even days of work by allowing me to debug locally instead of constantly recreating builds on Thunderhead until I get something working. Besides the problems as described above everything has been great! I hope my experience and challenges faced helps in making Thunderhead and the local debugger a better tool. I greatly appreciate you helping me to debug on my end and I look forward to a continuing relationship with PlayFab.