GeForce NOW SDK:
Cloud Check API

API Reference and Integration Guide

Document History

Version	Date	Description of Change
1.0	04/15/2022	First release
2.0	07/08/2022	Rework doc for new gfnIsRunningInCloud design
2.1	10/27/2022	Added note about in-stream patching
3.0	01/30/2024	Rework doc for new GfnCloudCheck design
3.1	06/26/2024	Removed notes on session streaming for CloudCheck call

Introduction

The Cloud Check API is a part of the GeForce NOW (GFN) SDK which allows application developers to check whether their applications are running on GFN.

Audience

- Publishers whose launcher or store applications may wish to alter their behavior if they are running on GFN.
- Game or application developers whose games or applications may wish to alter their behavior if they detect they are running on GFN.
- Security, anti-cheat or game integrity focused developers who may wish to alter the behavior of these security mechanisms in the context of the GFN environment.

Overview

This document contains a high-level overview of GFN Cloud Check APIs, as well as information concerning best practices when running on GFN.

Key Concepts

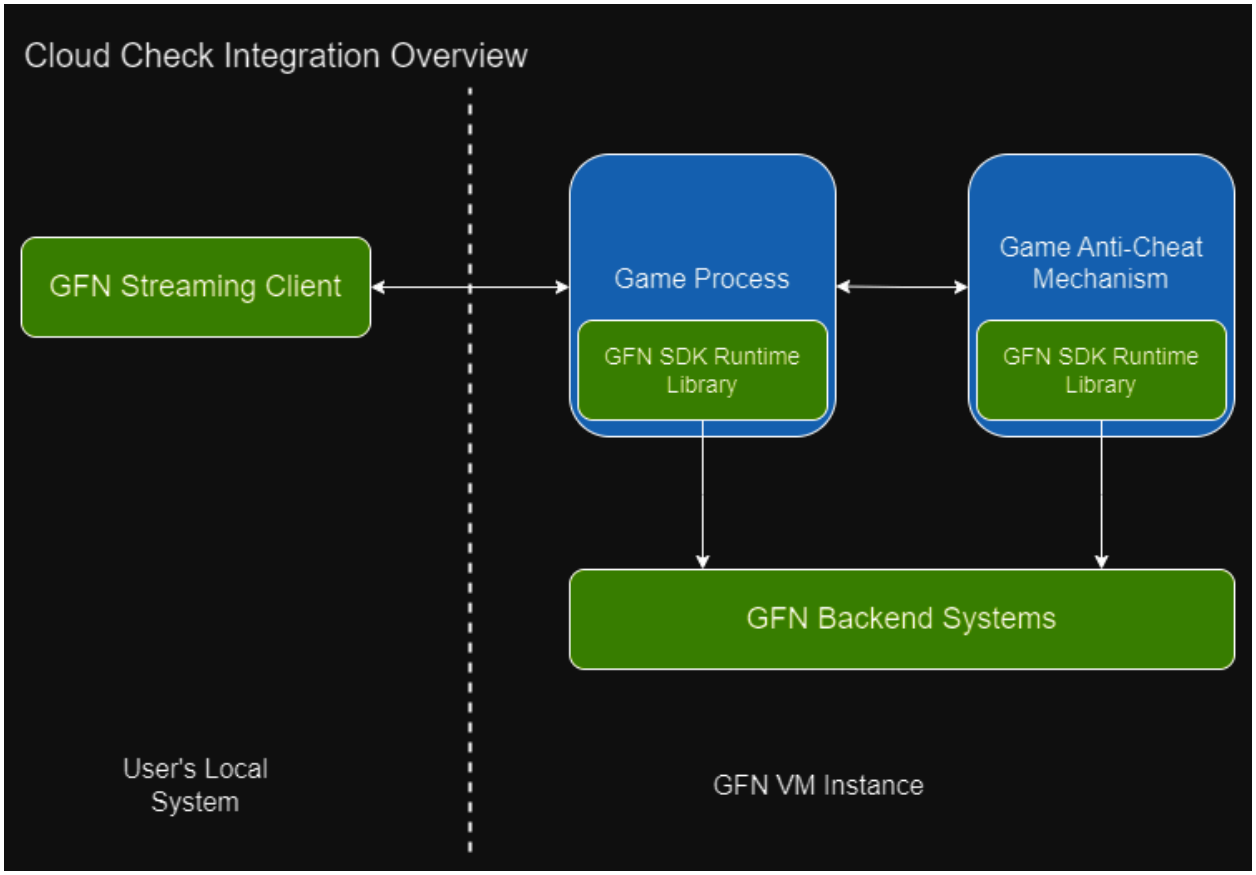
Game developers can use Cloud Check API to customize behavior or optimize performance of their games on GFN. The API can be used in different scenarios, for example:

1. Low risk scenarios where the game may decide to disable non-cloud UX.
2. Medium risk scenarios where the game may decide to provide GFN specific benefits (skin, levels, weapons, etc.).
3. High risk scenarios where the game may decide to implement anti-cheat optimizations.

Traditionally, games utilize anti-cheat software to prevent players from gaining an unfair advantage through third-party tools. These tools are employed when games are executed on a local system controlled by an end-user, wherein the state of the system may be manipulated to provide said user with an advantage in online games, to the detriment of other players.
Historically, game developers have employed various strategies to combat such scenarios, including the detection of abnormal play on the secure server hosting the game session and the identification of tampered files or rogue injections into the game process on the user's system.
However, when the same game is hosted on the GFN platform, the user's system and control over the game binaries and processes are eliminated from the equation. GFN's emphasis on security and tamper resistance renders it unlikely that the system can be compromised by the GFN user. Therefore, certain algorithms for detecting tampered files or rogue injections may be deemed unnecessary, as any potential performance impact they have on the game becomes superfluous as well.
Developers of these detection mechanisms now possess the capability to leverage the GFN SDK to securely query whether they are running in the GFN environment. This enables them to make a high-confidence decision to disable these mechanisms without risking negative impacts on other users.

To simplify integration and use of the API functions described in the document, the GFN SDK includes a C-based interface via a single source file, GfnRuntimeSdk_Wrapper.c, as well as a matching header file, GfnRuntimeSdk_Wrapper.h. These sources perform the heavy lifting of:

1. Dynamically loading the runtime component of the GFN SDK on GFN VMs after verifying its digital signature.
2. Calling appropriate functions defined in the GFN SDK Library.
3. Releasing the GFN SDK Library at the end of the API call.


Cloud Check Integration Overview
GFN Streaming Client
Game Process
GFN SDK Runtime Library
Game Anti-Cheat Mechanism
GFN SDK Runtime Library
GFN Backend Systems
User's Local System
GFN VM Instance

API Reference

The table below lists all wrapper methods related to the GFN Cloud Check API.

Method	Description
GfnCloudCheck	This method securely determines whether the application is running in the GFN environment. Optionally, it allows the caller to participate in the cryptographic cloud environment verification process.
GfnIsRunningInCloud	This method determines whether the application is running in the GFN environment.
GfnIsRunningInCloudSecure	This method is a legacy cloud check API, supported for backward compatibility only, and should not be used for new integrations.

This document describes the GFN Cloud Check API methods based on the wrapper definitions as wrapper provides the benefits of binary validation. The base definition of all methods is the same as wrapper methods, see GfnRuntimeSdk_CAPI.h header file for base definition.

GfnCloudCheck

The *GfnCloudCheck* API function is the newest Cloud Check method which employs a challenge-response protocol where the digitally signed response may optionally be validated on the game seat or by an external validation service of the application's choosing. Currently, this functionality is only available on Windows platform.

The *GfnCloudCheck* API function can be called to allow a game or application to determine whether to disable security features which may degrade performance:

- Turning off some or all costly anti-cheat features
- Disabling game or application binary integrity checks

The game or application itself may verify the cloud check service's response for added certainty that it is operating in a secure GFN environment.

Wrapper Interface

```
C/C++
GfnRuntimeError GfnCloudCheck(
[in] const GfnCloudCheckChallenge* challenge,
[out] GfnCloudCheckResponse* response,
[out] bool* isCloudEnvironment);
```

Parameters

`[in] const GfnCloudCheckChallenge* challenge`

```
C/C++
    typedef struct GfnCloudCheckChallenge
    {
       const char* nonce;
       unsigned int nonceSize;
    } GfnCloudCheckChallenge;
```

This is a pointer to a *GfnCloudCheckChallenge* structure. It is an optional input parameter to provide randomly generated nonce data. If a non-null challenge parameter is passed in, the

response parameter is mandatory. If the challenge parameter is null, a non-null response parameter will result in *gfnInvalidParameter* error.

`[out] const GfnCloudCheckResponse* response`

```
C/C++
    typedef struct GfnCloudCheckResponse
    {
       const char* attestationData;
       unsigned int attestationDataSize;
    } GfnCloudCheckResponse;
```

This is a pointer to a *GfnCloudCheckResponse* structure. It is an optional output parameter to receive digitally signed JWT attestation data that includes the challenge nonce. If a non-null response parameter is passed in, the challenge parameter is mandatory.

`[out] bool* isCloudEnvironment`

This is a pointer to a *bool.* It is an optional output parameter that receives true if the calling application is in the GFN environment.

Typical parameter usage:

1. The caller wishes to pass its own challenge data (nonce) and verify the digitally signed response itself, in which case challenge and response must be passed together.
2. The caller does not wish to perform additional validation, allowing the SDK to generate its own challenge data (nonce) and internally verify the response. In this case, the caller need only populate isCloudEnvironment.

Return value

`GfnRuntimeError`

Value	**Meaning**
gfnSuccess	The cloud check was performed, and the return parameters are valid.
gfnInvalidParameter	Indicates either a null challenge with non-null response or null response with non-null challenge was passed.

gfnNotAuthorized	Indicates the application is either not properly onboarded, or the application attempted to perform cloud check in an unsafe environment (the application has patched or otherwise altered files on the game seat).
gfnBackendError	Indicates the function could not communicate with the GFN backend services to confirm execution in a GFN environment.
gfnThrottled	Indicates the API call was throttled for exceeding the rate limit. Cloud check calls are rate controlled, so it is recommended that the result of the cloud check is cached by the calling process, if possible. The rate of cloud check calls should not exceed 3 attempts per minute, after which the call can return this error code.
gfnAPINotInit	Indicates GFN SDK API was not initialized. Prior to calling this function, ensure that the *GfnInitializeSdk* function is called first, and include any applicable partner ID provided by NVIDIA.
gfnAPINotFound	Indicates the function was not found in the GFN SDK Library.

Usage

1. GfnCloudCheck API call for low risk scenarios

The simple flavor of the GfnCloudCheck API call is intended for scenarios in which the calling application does not need to verify attestation data from the GFN SDK. The examples of low risk scenarios will be if the application wants to hide non-cloud UX, optimize startup sequence for GFN, or bypass the display driver checks from the application since GFN will guarantee the display driver is latest. The C code below demonstrates API call usage in these scenarios:

```
C/C++

bool SimpleGfnCloudCheck()

{
```

```
        bool isCloud = false;
        GfnError err = GfnCloudCheck(nullptr, nullptr, &isCloud);
        if (err != GfnError::gfnSuccess)
        {
                // Function failure means that it cannot be securely considered
                // GFN
                return false;
        }
        return isCloud;
}
```

2. GfnCloudCheck API call with attestation data validation

The C Code below shows how attestation data returned from the *GfnCloudCheck* API call can be locally verified on the game seat using platform native APIs. The nonce used in the *GfnCloudCheck* call is locally generated.

```
C/C++
#define NONCE_SIZE 16;
bool ValidatedGfnCheck()
{
        bool bCloudCheck = false;
        char nonce[NONCE_SIZE] = { 0 };
        if (!GfnCloudCheckGenerateNonce(nonce, _countof(nonce)))
        {
                return false;
        }
        else
        {
                GfnCloudCheckChallenge challenge = { nonce, _countof(nonce) };
                GfnCloudCheckResponse response = { nullptr, 0 };
```

```
            GfnError err = GfnCloudCheck(&challenge, &response, nullptr);
            if (err != GfnError::gfnSuccess)
            {
                  // Function failure means that it cannot be securely
                  // considered GFN
                  return false;
            }
            else
            {
                  if (response.attestationData != nullptr)
                  {
                        bCloudCheck =
                        GfnCloudCheckVerifyAttestationData(response.attestati
                        onData, challenge.nonce, challenge.nonceSize);
                        GfnFree(&response.attestationData);
                  }
                  else
                  {
                        bCloudCheck = false;
                  }
            }
      }
      return bCloudCheck;
}
```

As a part of the cloud check package the SDK offers template source code to validate the response JWT (See Appendix for details on JWT format). This source code is self-contained with no external dependencies. The code does the following:

1. Validates the digital signature and checks the certificate chain.
2. Matches the nonce.
3. Sanity-checks JWT fields for validity.

Reference

The GFN SDK offers platform-specific verification code designed to validate the JWT returned in the *GfnCloudCheckResponse* output parameter obtained from a *GfnCloudCheck* call. Developers have the flexibility to incorporate the source files of the validation code into their own builds. Currently, GfnCloudCheck API is only supported on Windows Platform. The reference code is developed using Windows native API.

Reference Files:
https://github.com/NVIDIAGameWorks/GeForceNOW-SDK/tree/master/samples/Common/GfnCloudCheckUtils.h

https://github.com/NVIDIAGameWorks/GeForceNOW-SDK/tree/master/samples/Common/Platform/Win/GfnCloudCheckUtils.c

GfnIsRunningInCloud

The *GfnIsRunningInCloud* API function is the first version of the cloud check.It makes fast queries to GFN systems to determine if the application is running in GFN. This is not a secure cloud check method. This API call can be used on Linux, since it is the only Cloud Check API function currently available on Linux.

Wrapper Interface

```
C/C++

GfnRuntimeError GfnIsRunningInCloud(

[out] bool* runningInCloud);
```

Parameters

`[out] bool* runningInCloud`

Value	Meaning
true	Running in GFN Cloud environment
false	Not running in the GFN Cloud environment, or an internal error was encountered during the function call.

Return value

`GfnRuntimeError`

Value	Meaning
gfnSuccess	The query was successful, and the return parameter is valid.
gfnAPINotInit	Indicates GFN SDK was not initialized. Prior to calling this function, ensure that the *GfnInitializeSdk* function is called first, and include any applicable partner ID provided by NVIDIA.

gfnAPINotFound	Indicates the function was not found in the GFN SDK Library.

GfnIsRunningInCloudSecure

GfnIsRunningInCloudSecure is a legacy API function supported for backward compatibility. This function has been internally modified to redirect to the *GfnCloudCheck* function. This function is a legacy cloud check API, supported for backward compatibility only, and should not be used for new integrations

Wrapper Interface

```
C/C++

GfnRuntimeError GfnIsRunningInCloudSecure(
[out] GfnIsRunningInCloudAssurance* assurance);
```

Parameters

`[out] GfnIsRunningInCloudAssurance* assurance`

This is a pointer to a *GfnIsRunningInCloudAssurance* enum value whose semantics are defined below:

Value	**Meaning**
gfnNotCloud	Not considered to be running in GFN cloud.
gfnIsCloudLowAssurance	Deprecated
gfnIsCloudMidAssurance	Determined to be running in GFN Cloud, using software and network heuristics that are difficult to circumvent.
gfnIsCloudHighAssurance	Deprecated

Return value

`GfnRuntimeError`

Value	Meaning
gfnSuccess	The query was successful, and the return parameter is valid.
gfnAPINotInit	Indicates GFN SDK was not initialized. Prior to calling this function, ensure that the *GfnInitializeSdk* function is called first, and include any applicable partner ID provided by NVIDIA.
gfnAPINotFound	Indicates the function was not found in the GFN SDK Library

Security Best Practices

Cloud Check API functions described in the section above are for use by software running in the context of the applications. GFN offers another API that can be used by applications' backend services to detect if the application is making a call to backend services is originating from GFN. See GFN IP API Guide for more details.

Appendix

The Response returned by *GfnCloudCheck* API contains attestation data formatted as JWT structure as documented in RFC 7519. It is a string that is built of three parts -

1. the header that describes the format of the data and the signature.
2. the data, which is a set of individual values called claims.
3. the signature, which is a digital signature to validate the integrity and authenticity of both the header and the data.

These parts are independently base64url encoded and concatenated into a single string separated by '.'(dot) characters.

Both the header and data are formatted in JSON, whereas the signature block is a base64url-encoded string computed over the concatenated string of the base64url-encoded header and data delimited by the '.' character. A symbolic representation of a properly formatted JWT, assuming the RS512 digital signature algorithm is used, is shown below:

base64url(header).base64url(data).base64url(RSASHA512(base64url(header).base64url(data)))

The GFN SDK fully validates the JWT before providing it to the caller in the *GfnCloudCheckResponse* output parameter populated by a call to *GfnCloudCheck*. If the caller wishes to perform their own validation, they must:

1. Verify the nonce:
   The nonce is one of the claims in the JWT data. To prevent replay attacks, the nonce must be the same one provided in the *GfnCloudCheckChallenge* data passed to *GfnCloudCheck* call.
2. Verify algorithm field:
   The JWT returned in the GfnCloudCheckResponse output parameter populated by a call to *GfnCloudCheck* is signed using the RS512 algorithm. The caller must verify that the algorithm field of the header is set to RS512.
3. Verify digital signature:
   The caller must verify the signature using the RS512 algorithm.

The table below details all steps that can be performed to validate the provided JWT.

Field Name	Location	Mandatory	Value	Details
x5c	Header	Yes	Valid certificate chain	The x5c field contains the certificate chain with the first element pointing to the leaf certificate. The caller must verify that the certificate chain is valid. The root certificate has to be pinned in the verifier code, as it is not a part of the chain.
alg	Header	Yes	RS512	Lofn currently uses RSASHA512, so the alg field must be set to RS512.

Field Name	Location	Mandatory	Value	Details
kid	Header	No	UUID	The kid field must be set to a UUID that was generated at the time of creation of the leaf certificate.
nonce	Data	Yes	Input nonce	The nonce field must match the nonce that was provided in the GfnCloudCheckChallenge struct passed to GfnCloudCheck.
	Signature	Yes	RSASHA512 signature	Digital signature over header and body validated using RSASHA512 algorithm.