NETWORK GAT INTERFACE SPECIFICATION V1.1

Gaming Standards Association

S2S Technical Committee

Released: 2019/07/11

Copyright Page

Document Title: Network GAT Interface Specification V1.1

Release Date: 2019/07/11, by Gaming Standards Association® (GSA®).

Patents and Intellectual Property

NOTE: The user's attention is called to the possibility that compliance with this [standard/specification] may require use of an invention covered by patent rights. By publication of this [standard/specification], GSA takes no position with respect to the validity of any such patent rights or their impact on this [standard/specification]. Similarly, GSA takes no position with respect to the terms or conditions under which such rights may be made available from the holder of any such rights. Contact GSA for further information.

Trademarks and Copyright

GSA Contact Information

E-mail: sec@gamingstandards.com

WWW: http://www.gamingstandards.com

Table of Contents

I About This Document

This document defines requirements that must be met by gaming devices, hosts, and messaging protocols to be compliant with the Network GAT Interface Specification.

I.I Acknowledgements

The Gaming Standards Association would like to express its appreciation to all members of the S2S committee, past and present, for their significant contribution and dedication to the creation of this specification.

I.II Document Conventions

I.II.I Indicating Requirements, Recommendations, and Options

Terms and phrases in this document that indicate requirements, recommendations, and options in the Network GAT Interface Specification are used as defined in the IETF RFC 2119.

In summary:

Requirements:

To indicate requirements, this document uses "MUST", "MUST NOT", "REQUIRED", "SHALL", or "SHALL NOT".

Recommendations:

To indicate recommendations, this document uses "SHOULD", "SHOULD NOT", "RECOMMENDED".

Options:

To indicate **options**, this document uses "MAY" or "OPTIONAL".

I.II.II Other Formatting Conventions

- Blue text indicates an internal link or external hyperlink to a URL.
- **Bold** (other than in headings) or underlined text is used for emphasis, unless specifically indicated otherwise.
- `Courier New` font is used to indicate JSON components and their values, and other types of code or pseudo code.

I.III Categorization of Standards

To help provide guidance to implementers regarding the maturity and stability of its standards, GSA categorizes its standards as Candidate Standards, Proposed Standards, or Recommended Standards. The categorizations can be found on the GSA website.

- Standards identified as Candidate Standards are the least mature; changes to these standards should be expected in future releases.
- Standards identified as Proposed Standards have been reduced to practice and deployed; very few changes to these standards should be expected.

- Standards identified as Recommended are the most mature and have been widely deployed; no changes to these standards should be expected.

Further details about the categorization of standards and extensions can be found in the GSA Policy Handbook.

Chapter 1

Network GAT Interface

Overview

1.1 Introduction

This document describes the Network GAT Interface, which contains commands specifically designed to provide software authentication information required by gaming regulators.

The Network GAT Interface allows a GAT Client to communicate with a GAT Server and request that the GAT Server perform a verification algorithm (such as a hashing algorithm) on components exposed by the GAT Server or on components exposed by end-clients of the GAT server. These components may include files, executables, loadable modules (such as DLLs), or downloadable content (such as game packages, SWFs, etc.)

The interface gives systems a standard method for reporting the resulting verification data to regulators and other systems. The verification data can be compared to expected results that have been gathered from a GAT Trusted Source or that are stored in a local database on the GAT Client.

1.1.1 Terminology

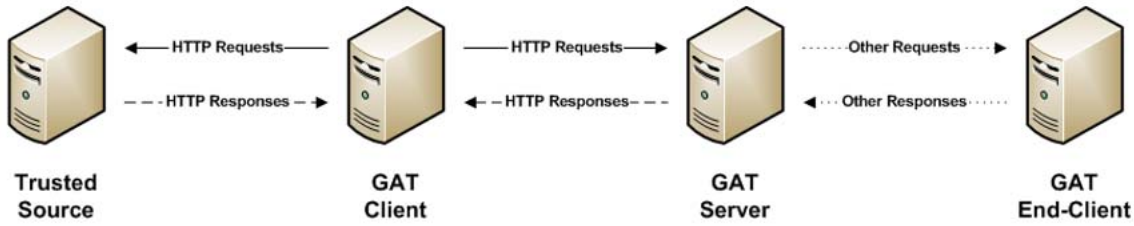
Within this document, the following definitions are used when referring to GAT Clients, GAT Servers, and GAT Trusted Sources.

Table 1.1 Terminology

	Description
GAT Client	The GAT Client represents the system that wishes to challenge the GAT Server to perform verifications upon exposed components.
GAT End-Client	A GAT End-Client is an end-point that is communicating with the GAT Server. The GAT Server may expose components of GAT End-Clients, as well as its own components, to a GAT Client. The GAT Server may be communicating with the GAT End-Client using NGI or some other protocol.
GAT Server	The GAT Server is the system that exposes components that can be verified by a remote GAT Client. The GAT Server may expose executables, DLLs, downloadable content (such as packages that can be downloaded by EGMs and modules that represent those installed packages), or other types of content (such as SWFs). Those components may be installed on the GAT Server itself or on GAT End-Clients communicating with the GAT Server.
GAT Trusted Source	The GAT Trusted Source represents an optional system that exposes expected GAT results for verifiable components. If the GAT Client does not know whether it can trust the results from the GAT Server, it can query the GAT Trusted Source for the expected GAT results and then use that information to verify that the GAT results from the GAT Server are correct.

These relationships are illustrated in the following diagram:

Figure 1.1 Network GAT Interface Overview



1.1.2 HTTP Verb Conventions

The following table lists the standard conventions followed in this document. These conventions may not apply to every scenario. Please consult the individual resources in the following sections for the specific rules that apply.

Table 1.2 HTTP Verbs

Resource	POST create	GET read	PUT update	DELETE delete
/anyResource	Create a new item in the resource	Search the resource	Update multiple items	Delete multiple items
/anyResource/[ItemID]	Return Error - ItemID may only be assigned by the server	If exists, return ItemID; else, return Error	If exists, update ItemID; else, return Error	If exists, delete ItemID; else, return Error

Chapter 2

GAT Server Supported HTTP Verbs

2.1 Introduction

The following table lists the resources that MUST be exposed by a GAT Server, along with additional features and actions that can be performed on those resources. For additional information, please refer to the sections describing each individual resource that follow.

Table 2.1 HTTP Verbs

HTTP Verbs						
Resource	**POST (create)**	**GET (read)**	**PUT (update)**	**DELETE (delete)**	**Additional Features**	**Actions**
/gat/[ver]/components	No	Yes	No	No	Search	
/gat/[ver]/components/[ID]	No	Yes	No	No	Search	
/gat/[ver]/verificationTasks	Yes	Yes	No	No	Search	Verify
/gat/[ver]/verificationTasks/[ID]	No	Yes	No	No		
/gat/[ver]/verificationSearch	Yes	Yes	No	No		Verify
/gat/[ver]/verificationSearch/[ID]	No	Yes	No	No		

The [ver] segment of the resource MUST be replaced with the version number of the Network GAT Interface. A dot-delimited format MUST be used for the version number. For example, version "1.0" must be represented as the character string "1.0". Superfluous leading and trailing characters MUST be removed. For example, version "v1.0" must be represented as "1.0". For version "1.1", the appropriate resources are:

Table 2.2 Version 1.1 Resources

Version "1.1" Resource
/gat/1.1/components
/gat/1.1/components/[ID]
/gat/1.1/verificationTasks
/gat/1.1/verificationTasks/[ID]
/gat/1.1/verificationSearch
/gat/1.1/verificationSearch/[ID]

2.2 Component Resources

A component is a software module, such as a server-side executable, DLL, image, or downloadable content, such as a game image, SWF, etc. The GAT Server MUST expose components that are subject to regulatory approval. Resources within this section are used to report individual components, or lists of components, that can be remotely authenticated by the GAT Client based upon specific search criteria.

2.2.1 Component Details

This resource returns information about a single component. The component is specified in the [ID] segment of the resource.

2.2.1.1 Resource Information

Table 2.3 Resource Information

HTTP Method	GET
URI	/gat/[ver]/components/[ID], where [ID] is the identifier of the component
Request Headers	Accept: application/json
Response Content Type	application/json
Response Content	component Object. See Section 2.2.1.4, Request Errors below for more detail about error scenarios.

2.2.1.2 Request Parameters

The `endClientType` and `endClientID` parameters MUST both be included in the request or MUST both be omitted from the request. When included, information about the component on the specified GAT End-Client MUST be included in the response or an appropriate HTTP error code MUST be returned. When omitted, information about the component on the GAT Server MUST be included in the response or an appropriate HTTP error code MUST be returned.

Table 2.4 Component Resources - Request Parameters

Parameter Name	Type	Optional or Required	Description
`endClientType`	String	Optional	End Client Type; type of GAT End-Client; see Section 4.1.3, endClientType Enumerations for permitted values.
`endCliendId`	String	Optional	End Client ID; identifier of the GAT end-Client.

2.2.1.3 Response

The `endClientType` and `endClientID` parameters MUST both be included in the response or MUST both be omitted from the response. If information about the component on a GAT End-Client is being reported, the parameters MUST be included. If information about the component on the GAT Server is being reported, the parameters MUST be omitted.

Table 2.5 component Object

Field	Type	Optional or Required	Description
`endClientType`	String	Optional	End Client Type; type of GAT End-Client; see Section 4.1.3, endClientType Enumerations for permitted values.
`endClientId`	String	Optional	End Client ID; identifier of the GAT End-Client.
`componentID`	String	Required	The component identifier of the resource.
`componentType`	String	Optional	String enumeration that identifies the type of component. See Table 4.1 for more information.
`description`	String	Optional	Contains an optional description of the component.
`size`	Number	Optional	The number of bytes for this component. Specifying this allows the client to generate its own offsets when requesting verification. This field MUST be sent if the component supports an algorithm which supports offsets.
`algorithms`	Array of algorithm objects	Required	At least one algorithm MUST be supported.

Table 2.6 algorithm Object

Field	Type	Optional or Required	Description
`algorithmType`	String	Required	Uniquely identifies the verification algorithm. See Table 4.2 for more information.
`supportsSeed`	Boolean	Required	Indicates whether the verification algorithm supports a seed (typically used by checksums).
`supportsSalt`	Boolean	Required	Indicates whether the verification algorithm supports a salt (arbitrary bytes added to the component).
`supportsOffset`	Boolean	Required	Indicates whether the verification algorithm supports offsetting.

2.2.1.4 Request Errors

Table 2.7 Request Errors

HTTP Error Code	Description
`HTTP 404`	Resource not found

2.2.2 Component Search

This resource returns an array of components based on search parameters presented by the GAT Client.

2.2.2.1 Resource Information

Table 2.8 Resource Information

HTTP Method	GET
URI	/gat/[ver]/components
Request Headers	Accept: application/json
Response Content Type	application/json
Response Content	components Object. See Section 2.2.2.4, Request Errors below for more detail about error scenarios.

2.2.2.2 Request Parameters

The `endClientType` and `endClientID` parameters MUST both be included in the request or MUST both be omitted from the request. When included, information about components on the specified GAT End-Client MUST be included in the response or an appropriate HTTP error code MUST be returned. When omitted, information about components on the GAT Server MUST be included in the response or an appropriate HTTP error code MUST be returned.

The wildcard `NGI_all` MAY be used with the `endClientType` and `endClientID` parameters. When `NGI_all` is specified for the `endClientType` parameter, information about components for all GAT End-Clients MUST be included in the response regardless of the type of end-client — that is, filtering on the `endClientType` MUST NOT be performed. Similarly, when `NGI_all` is specified for the `endClientID` parameter, information about components for all GAT End-Clients MUST be included in the response regardless of the identifier of the end-client — that is, filtering on the `endClientID` MUST NOT be performed.

Table 2.9 Request Parameters

Parameter Name	Type	Optional or Required	Description
`endClientType`	String	Optional	End Client Type; type of GAT End-Client; wildcard `NGI_all` is permitted; see Section 4.1.3, endClientType Enumerations for permitted values.
`endClientId`	String	Optional	End Client ID; identifier of the GAT End-Client; wildcard `NGI_all` is permitted.
`includeSoftware`	Boolean	Optional	Set to true if software components SHOULD be included in the response. Set to false if software components SHOULD NOT be included in the response. If this parameter is not included in the request, a value of true MUST be assumed.
`includeOS`	Boolean	Optional	Set to true if OS components SHOULD be included in the response. Set to false if OS components SHOULD NOT be included in the response. If this parameter is not included in the request, a value of true MUST be assumed.

Table 2.9 Request Parameters

Parameter Name	Type	Optional or Required	Description
`includeContent`	Boolean	Optional	Set to true if content, such as content for game themes, SHOULD be included in the response. Set to false if content components SHOULD NOT be included in the response. If this parameter is not included in the request, a value of true MUST be assumed.

2.2.2.3 Response

Table 2.10 components Object

Field	Type	Optional or Required	Description
`components`	Array of component objects	Required	At least one component

2.2.2.4 Request Errors

Table 2.11 Request errors

HTTP Error Code	Description
`HTTP 404`	No resources found

2.3 Verification Task Resources

Resources within this section allow component verification tasks to be created on the GAT Server and provide access to the status of those verification tasks. Verification tasks can have various states, such as pending, in process, or complete. After verification tasks are complete, the final value of the verification request, called the verification result, will be reported.

2.3.1 Create Verification Task

This action creates a component verification task for the specified component.

2.3.1.1 Resource Information

Table 2.12 Resource Information

HTTP Method	POST
URI	/gat/[ver]/verificationTasks
Request Headers	Accept: application/json
Response Content Type	application/json
Response Content	HTTP 201 response with the HTTP Location Header set to the location of the newly created verification task. See Section 2.3.1.4, Request Errors below for more detail about error scenarios.

2.3.1.2 Request Parameters

The `endClientType` and `endClientID` parameters MUST both be included in the request or MUST both be omitted from the request. When included, a verification task for the specified component on the specified GAT End-Client MUST be created or an appropriate HTTP error code MUST be returned. When omitted, a verification task for the specified component on the GAT Server MUST be created or an appropriate HTTP error code MUST be returned.

Table 2.13 Request Parameters

Parameter Name	Type	Optional or Required	Description
`endClientType`	String	Optional	End Client Type; type of GAT End-Client; see Section 4.1.3, endClientType Enumerations for permitted values.
`endClientID`	String	Optional	End Client ID; identifier of the GAT End-Client.
`componentID`	String	Required	The `componentID` of the component on which this verification task is operating.
`algorithmType`	String	Required	Which authentication algorithm to run on the component.
`seed`	Number	Optional	The seed for the algorithm. Certain algorithms, such as checksums (CRC), use a seed to define the starting value.

Table 2.13 Request Parameters

Parameter Name	Type	Optional or Required	Description
`salt`	String	Optional	Arbitrary bytes that are prefixed to the component's byte buffer before the hash is generated (and after the offsets are applied)
`startOffset`	Number	Optional	The starting offset for the verification component
`endOffset`	Number	Optional	The ending offset for the verification component; if set to `-1` (negative one), indicates that no offset is specified, which is equivalent to specifying the index of the final valid byte of the buffer

2.3.1.3 Response

Table 2.14 Response

HTTP Response Code	Description
`HTTP 201 "Created"`	The HTTP Location Header is set to the location of the newly created verification task.

2.3.1.4 Request Errors

Table 2.15 Request Errors

HTTP Error Code	Description
`HTTP 400 "Bad Request"`	The `seed`, `salt`, `startOffset`, or `endOffset` is invalid for the component and/or algorithm.
`HTTP 404 "Not Found"`	The resource the client attempted to perform a verification on does not exist.
`HTTP 405 "Method Not Allowed"`	The algorithm type requested is not supported for the component.
`HTTP 413 "Request Entity Too Large"`	The `salt` requested is too large for the web server to process.

2.3.2 Verification Task Details

This resource returns the status and the final result of a verification task. Verification tasks have various states, such as pending, in process, or complete. The verification task is specified in the [ID] segment of the resource. The identifier for the verification task is assigned by the GAT Server when the verification task is created.

The result of the verification is contained in the `verifyResult` field, which is represented as a string. Thus, unless the verification algorithm explicitly specifies another format, the result MUST be converted from its native binary format into a hexadecimal character representation. Each 8-bit byte MUST be converted into two hexadecimal characters in big-endian order using the ASCII characters 0-9 and A-F. The resulting string MUST be included in the `verifyResult` field included in the Verification Task resource response.

The server is allowed to remove verification task resources over time to free up resources (database, or otherwise). The schedule for removing verification tasks is up to the server to determine. However, failed and completed verification task resources SHOULD be removed every 48 hours.

2.3.2.1 Resource Information

Table 2.16 Resource Information

HTTP Method	GET
URI	/gat/[ver]/verificationTasks/[ID], where [ID] is the identifier of the verification task
Request Headers	Accept: application/json
Response Content Type	application/json
Response Content	verificationTask Object. See Section 2.3.2.4, Request Errors below for more detail about error scenarios.

2.3.2.2 Request Parameters:

There are no additional request parameters.

2.3.2.3 Response Information:

The `endClientType` and `endClientID` parameters MUST both be included in the response or MUST both be omitted from the response. If information about a verification task on a GAT End-Client is being reported, the parameters MUST be included. If information about a verification task on the GAT Server is being reported, the parameters MUST be omitted.

Table 2.17 verificationTask Object

Field	Type	Optional or Required	Description
`verificationID`	Number	Required	The verification ID of this verification task.
`endClientType`	String	Optional	End Client Type; type of GAT End-Client; see Section 4.1.3, endClientType Enumerations for permitted values.
`endClientID`	String	Optional	End Client ID; identifier of the GAT End-Client.
`componentID`	String	Required	The `componentID` of the component on which this verification task is operating
`verifyState`	String	Required	The current state of the verification task. Allowed response values are `NGI_queued`, `NGI_inProcess`, `NGI_complete` and `NGI_error`.
`algorithmType`	String	Required	Which authentication algorithm to run on the component

Table 2.17 verificationTask Object

Field	Type	Optional or Required	Description
`verifyResult`	String	Optional	The verification result; not reported unless the `verifyState` is set to `NGI_complete`

2.3.2.4 Request Errors

Table 2.18 Request Errors

HTTP Error Code	Description
`HTTP 404`	Resource not found

2.3.3 Verification Task Search

This resource returns an array of verification tasks based on search parameters presented by the GAT Client.

2.3.3.1 Resource Information

Table 2.19 Resource Information

HTTP Method	GET
URI	/gat/[ver]/verificationTasks
Request Headers	Accept: application/json
Response Content Type	application/json
Response Content	`verificationTasks` Object

2.3.3.2 Request Parameters

The `endClientType` and `endClientID` parameters MUST both be included in the request or MUST both be omitted from the request. When included, information about verification tasks for the specified component on the specified GAT End-Client MUST be included in the response or an appropriate HTTP error code MUST be returned. When omitted, information about verification tasks for the specified component on the GAT Server MUST be included in the response or an appropriate HTTP error code MUST be returned.

The wildcard `NGI_all` MAY be used with the `endClientType` and `endClientID` parameters. When `NGI_all` is specified for the `endClientType` parameter, information about verification tasks for all GAT End-Clients MUST be included in the response regardless of the type of end-client — that is, filtering on the `endClientType` MUST NOT be performed. Similarly, when `NGI_all` is specified for the `endClientID` parameter, information about verification tasks for all GAT End-Clients MUST be included in the response regardless of the identifier of the end-client — that is, filtering on the `endClientID` MUST NOT be performed.

Table 2.20 Request Parameters

Parameter Name	Type	Optional or Required	Description
`endClientType`	String	Optional	End Client Type; type of GAT End-Client; wildcard `NGI_all` is permitted. See Section 4.1.3, endClientType Enumerations for permitted values.
`endClientID`	String	Optional	End Client ID; identifier of the GAT End-Client; wildcard `NGI_all` is permitted.
`componentID`	String	Required	The `componentID` of the component targeted for a verification task search.

2.3.3.3 Response

Table 2.21 verificationTasks Object

Field	Type	Optional or Required	Description
`verificationTasks`	Array of `verificationTask` Objects	Required	An array of verification tasks that matched the requested component ID.

2.3.3.4 Request Errors

Table 2.22 Request Errors

HTTP Error Code	Description
`HTTP 400 "Bad Request"`	The component ID requested does not exist.
`HTTP 404 "Not Found"`	No verification tasks exist for the requested resource.

2.4 Verification Search Resources

Resources within this section allow verification requests to be created on the GAT Server and provide access to the status of those verification requests. Verification requests can have various states, such as pending, in process, or complete. After verification requests are complete, the final value of the verification requests, called the verification results, will be reported.

Resources described within this section are similar to the Verification Task Resources described above. The primary difference is that resources described in this section can be used to submit a single request to verify multiple components. The Verification Task Resources described above only allow one component to be specified in a request.

The request to verify multiple components is modeled after the Component Search resource described above. Rather than specifying a specific component, the GAT Client specifies the type of components that it wants verified. The GAT Server is responsible for identifying the specific components meeting those criteria. The Verification Task Resources described above should be used to verify individual components.

2.4.1 Create Verification Search

This action creates a Verification Search for the types of components specified in the request. The GAT Server is responsible for selecting the specific components meeting the request parameters and then verifying those components using the specified algorithm, salt, seed, and offset values. The fully expanded list of components MUST be included when the Verification Search Details resource is requested.

2.4.1.1 Resource Information

Table 2.23 Resource Information

HTTP Method	POST
HTTP Method	POST
URI	/gat/[ver]/verificationSearch
Request Headers	Accept: application/JSON
Response Content Type	application/JSON
Response Content	HTTP 201 response with the HTTP Location Header set to the location of the newly created Verification Search. See Section 2.4.1.4, Request Errors below for more detailed about error scenarios.

2.4.1.2 Request Parameters

The `endClientType` and `endClientID` parameters MUST both be included in the request or MUST both be omitted from the request. When included, a Verification Search for the specified component on the specified GAT End-Client MUST be created or an appropriate HTTP error code MUST be returned. When omitted, a Verification Search for the specified component on the GAT Server MUST be created or an appropriate HTTP error code MUST be returned.

Table 2.24 Request Parameters

Parameter Name	Type	Usage	Description
`endClientType`	String	Optional	End Client Type; type of GAT End-Client; see Section 4.1.3, endClientType Enumerations for permitted values.
`endClientId`	String	Optional	End Client ID; identifier of the GAT End-Client.
`includeSoftware`	Boolean	Optional	Set to true if software components SHOULD be verified. Set to false if software components SHOULD NOT be verified. If this parameter is not included in the request, a value of true MUST be assumed.
`includeOS`	Boolean	Optional	Set to true if OS components SHOULD be verified. Set to false if OS components SHOULD NOT be verified. If this parameter is not included in the request, a value of true MUST be assumed.
`includeContent`	Boolean	Optional	Set to true if content, such as content for game themes, SHOULD be verified. Set to false if content components SHOULD NOT be verified. If this parameter is not included in the request, a value of true MUST be assumed.
`algorithmType`	String	Required	Indicates which authentication algorithm to use to verify the selected components.
`seed`	Number	Optional	The seed for the algorithm. Certain algorithms, such as checksums (CRC), use a seed to define the starting value for the calculation.
`salt`	String	Optional	Arbitrary bytes that are prefixed to the component's byte buffer before the hash is generated (and after the offsets are applied).
`startOffset`	Number	Optional	The starting offset within each component.
`endOffset`	Number	Optional	The ending offset within each component; if set to -1 (negative one), indicates that no offset is specified, which is equivalent to specifying the index of the final valid byte of the buffer.

2.4.1.3 Response Information

Table 2.25 Response Information

HTTP Response Code	Description
`HTTP 201 "Created"`	The HTTP Location Header is set to the location of the newly created Verification Search.

2.4.1.4 Request Errors

Table 2.26 Request Errors

HTTP Error Code	Description
`HTTP 400 "Bad Request"`	The `seed`, `salt`, `startOffset`, or `endOffset` is invalid for the selected components and/or algorithm.
`HTTP 404 "Not Found"`	No components of the specified types exist.
`HTTP 405 "Method Not Allowed"`	The algorithm type requested is not supported for the selected components.
`HTTP 413 "Request Entity Too Large"`	The salt requested is too large for the web server to process.

2.4.2 Verification Search Details

This resource returns the status and the final result of a Verification Search request. Requests have various states, such as pending, in process, or complete. The Verification Search is specified in the `[ID]` segment of the resource. The identifier for the Verification Search is assigned by the GAT Server when the Verification Search is created.

The verification result is contained in the `verifyResult` field, which is represented as a string. Thus, unless the verification algorithm explicitly specifies another format, the result MUST be converted from its native binary format into a hexadecimal character representation. Each 8-bit byte MUST be converted into two hexadecimal characters in big-endian order using the ASCII characters 0-9 and A-F. The resulting string MUST be included in the `verifyResult` field included in the response.

The GAT Server is allowed to remove Verification Search resources over time to free up resources (database, or otherwise). The schedule for removing Verification Searches is up to the server to determine. However, failed and completed Verification Searches resources SHOULD be removed every 48 hours.

2.4.2.1 Resource Information

Table 2.27 Resource Information

HTTP Method	GET
URI	`/gat/[ver]/verificationSearch/[ID]`, where `[ID]` is the identifier of the Verification Search.
Request Headers	Accept: `application/JSON`

Table 2.27 Resource Information

HTTP Method	GET
Response Content Type	`application/JSON`
Response Content	`verificationSearch` Object. See Section 2.4.1.4, Request Errors below for more detail about error scenarios.

2.4.2.2 Request Parameters

There are no additional request parameters.

2.4.2.3 Response Information

The `endClientType` and `endClientID` parameters MUST both be included in the response or MUST both be omitted from the response. If information about a Verification Search on a GAT End-Client is being reported, the parameters MUST be included. If information about a Verification Search on the GAT Server is being reported, the parameters MUST be omitted.

Table 2.28 verificationSearch Object

Parameter Name	Type	Usage	Description
`verificationId`	Number	Required	The verification ID of the Verification Search; assigned by the GAT Server.
`endClientType`	String	Optional	End Client Type; type of GAT End-Client; see Section 4.1.3, endClientType Enumerations for permitted values.
`endClientId`	String	Optional	End Client ID; identifier of the GAT End-Client.
`includeSoftware`	Boolean	Optional	Set to the corresponding value from the Verification Search request.
`includeOS`	Boolean	Optional	Set to the corresponding value from the Verification Search request.
`includeContent`	Boolean	Optional	Set to the corresponding value from the Verification Search request.
`algorithmType`	String	Required	Set to the corresponding value from the Verification Search request.
`componentResults`	Array	Required	Array of `componentResult` objects.

Table 2.29 componentResult Object

Parameter Name	Type	Usage	Description
`componentId`	String	Required	The component ID of the selected component.

Table 2.29 componentResult Object

Parameter Name	Type	Usage	Description
`verifyState`	String	Required	The current state of the verification request for the selected component. Allowed response values are `NGI_queued`, `NGI_inProcess`, `NGI_complete` and `NGI_error`.
`verifyResult`	String	Optional	The verification result; not reported unless the `verifyState` property is set to `NGI_complete`.

Table 2.30 Request Errors

HTTP Error Code	Description
`HTTP 404 "Not Found"`	Resource not found.

Chapter 3

GAT Trusted Source

Supported HTTP Verbs

3.1 Introduction

The following table lists the resources that MUST be exposed by a GAT Trusted Source, along with additional features and actions that can be performed on those resources. For additional information, please refer to the sections describing each individual resource that follow.

Table 3.1 HTTP Verbs

HTTP Verbs						
Resource	**POST (create)**	**GET (read)**	**PUT (update)**	**DELETE (delete)**	**Additional Features**	**Actions**
/gat/[ver]/expectedResults/[ID]	No	Yes	No	No		

The [ver] segment of the resource MUST be replaced with the version of the Network GAT Interface. A dot-delimited format MUST be used for the version. For example, version "1.0" must be represented as the character string "1.0". Superfluous leading and trailing characters MUST be removed. For example, version "v1.0" must be represented as "1.0". For version "1.1", the appropriate resources are:

Table 3.2 Version 1.1 Resources

Version "1.1" Resource
/gat/1.1/expectedResults/[ID]

3.2 Expected Results Resources

Resources within this section allow the expected results of component verification tasks to be requested from a GAT Trusted Source.

3.2.1 Expected Results Details

This resource provides expected results for the verification of a specified component.

3.2.1.1 Resource Information

Table 3.3 Resource Information

HTTP Method	GET
URI	/gat/[ver]/expectedResults/[ID], where [ID] is the identifier of the component
Request Headers	Accept: application/json
Response Content Type	application/json
Response Object	`expectedResults` Object. See Section 3.2.1.4, Request Errors below for more detail about error scenarios.

3.2.1.2 Request Parameters

Table 3.4 Request Parameters

Parameter Name	Type	Optional or Required	Description
`algorithmType`	String	Required	Algorithm type for which the expected results are required.
`supportsSeed`	Boolean	Required	Indicates whether a seed value should be provided for the expected result.
`supportsSalt`	Boolean	Required	Indicates whether a salt value should be provided for the expected result.
`supportsOffsets`	Boolean	Required	Indicates whether offsets should be provided for the expected result.

3.2.1.3 Response

Table 3.5 expectedResults Object

Field	Type	Optional or Required	Description
`componentID`	String	Required	Component Identifier; set to the `componentID` from the request.

Table 3.5 expectedResults Object

Field	Type	Optional or Required	Description
`algorithmType`	String	Required	Algorithm type; set to the `algorithmType` from the request.
`expectedResultList`	Array of `expectedResult` objects	Required	At least one expected result MUST be reported.

Table 3.6 expectedResult Object

Field	Type	Optional or Required	Description
`seed`	Number	Optional	Seed value to be used with the algorithm.
`salt`	String	Optional	Salt value to be used with the algorithm.
`startOffset`	Number	Optional	Starting offset to be used with the algorithm.
`endOffset`	Number	Optional	Ending offset to be used with the algorithm.
`expectedResult`	String	Required	Expected result from the algorithm.
`resultVersion`	String	Required	Identifies the specific version of the component associated with the expected result.
`resultStatus`	Boolean	Required	Indicates whether the expected result is approved (true) or revoked (false).
`expiration`	DateTime	Required	Date/time after which the expected results should no longer be used.

3.2.1.4 Request Errors

Table 3.7 Request Errors

HTTP Error Code	Description
`HTTP 400 "Bad Request"`	The `seed`, `salt`, `startOffset`, or `endOffset` is invalid for the component and/or algorithm.
`HTTP 404 "Not Found"`	Resource not found.
`HTTP 405 "Method Not Allowed"`	The algorithm type requested is not supported for the component.

Chapter 4

Data Types

4.1 Data Types

4.1.1 componentType Enumerations

Table 4.1 Enumerations of the componentType field

Enumeration	Description
NGI_software	A generic software component
NGI_OS	A component representing the operating system
NGI_content	A component representing content that is downloaded to a client, typically representing thin client content, such as image files, markup (HTML), or scripts.

4.1.2 algorithmType Enumerations

Table 4.2 Enumerations of the algorithmType field

Enumeration	Description
NGI_CRC16	A 16-bit cyclic redundancy check. Uses a 16-bit unsigned seed.
NGI_CRC32	A 32-bit cyclic redundancy check. Uses a 32-bit unsigned seed.
NGI_MD5	MD5 algorithm per RFC 1321. Seed is not supported, use salt.
NGI_SHA1	SHA-1 algorithm per FIPS 180-2. Seed is not supported, use salt.
NGI_SHA256	SHA-256 algorithm per FIPS 180-2. Seed is not supported, use salt.
NGI_SHA384	SHA-384 algorithm per FIPS 180-2. Seed is not supported, use salt.
NGI_SHA512	SHA-512 algorithm per FIPS 180-2. Seed is not supported, use salt.
NGI_HMACSHA1*	HMAC SHA-1 algorithm per FIPS 180-2 and FIPS 198. The HMAC key MUST be included as if it was a salt; seed and offsets are not supported.
NGI_HMACSHA256	HMAC SHA256 algorithm per FIPS 180-2 and FIPS 198. The HMAC key MUST be included as if it was a salt; seed and offsets are not supported.
NGI_HMACSHA512	HMAC SHA512 algorithm per FIPS 180-2 and FIPS 198. The HMAC key MUST be included as if it was a salt; seed and offsets are not supported.
NGI_SHA3_256	SHA3-256 algorithm per FIPS 202. Seed is not supported, use salt.
NGI_SHA3_512	SHA3-512 algorithm per FIPS 202. Seed is not supported, use salt.
NGI_HMACSHA3_256	HMAC SHA3-256 algorithm per FIPS 202 and FIPS 198. The HMAC key MUST be included as if it was a salt; seed and offsets are not supported.
NGI_HMACSHA3_512	HMAC SHA3-512 algorithm per FIPS 202 and FIPS 198. The HMAC key MUST be included as if it was a salt; seed and offsets are not supported.

*This algorithm is compatible with the HMAC SHA-1 algorithm required by the serial game authentication protocol GAT v3.50. It is intended that implementations using this algorithm be compatible with implementations of the HMAC SHA-1 algorithm in GAT v3.50. See GAT v3.50 for more details on the use of the HMAC SHA-1 algorithm. Note that the "(none)" convention, which is used in GAT v3.50, is not used in this specification.

4.1.3 endClientType Enumerations

Table 4.3 endClientType Enumerations

Enumeration	Description
`NGI_egm`	Electronic gaming machine.
`NGI_kiosk`	Self-service kiosk.
`NGI_cashier`	Cashier or other employee workstation.
`NGI_table`	Gaming table.
`NGI_sign`	Sign or other information display.
`NGI_bingo`	System managing a multi-player bingo operation.
`NGI_keno`	System managing a multi-player keno operation.
`NGI_fixedOdds`	Fixed odds betting system.
`NGI_parimutuel`	Parimutuel wagering system.
`NGI_system`	Peer system.
`NGI_all`	All types of end-clients; MUST only be used when specifically permitted within this specification.

NEW CHAPTER

Chapter 5

Event Codes

NEW CHAPTER

5.1 Event Code Table

The following table contains the list of event codes that can be used to report activity related to the Network GAT Interface. The Network GAT Interface does not have a native event reporting mechanism. When required, it is expected that these event codes will be reported through another protocol, such as GSA's Simple System Interface.

Table 5.1 Event Code Table

Event Code	Description
NGI_GAE100	Communications Established.
NGI_GAE105	Signature Authentication Successful.
NGI_GAE106	Signature Authentication Failure.

Chapter 6

Sample Messages

6.1 Component Details

Request:

```
GET /gat/1.1/components/ABC_123
```

Response:

```
{
    "componentId": "ABC_123",
    "componentType": "NGI_software",
    "description": "The ABC123 Component",
    "size": 0,
    "algorithms": [
        {
            "algorithmType": "NGI_SHA1",
            "supportsSeed": false,
            "supportsSalt": true,
            "supportsOffset": false
        },
        {
            "algorithmType": "NGI_HMACSHA1",
            "supportsSeed": false,
            "supportsSalt": true,
            "supportsOffset": false
        }
    ]
}
```

```
Request:

GET /gat/1.1/components/ABC_123?endClientType=NGI_kiosk&endClientID=ABC_12345

Response:

{
    "endClientType": "NGI_kiosk",
    "endClientID": "ABC_12345",
    "componentID": "ABC_123",
    "componentType": "NGI_software",
    "description": "The ABC123 Component",
    "size": 0,
    "algorithms": [
        {
            "algorithmType": "NGI_SHA1",
            "supportsSeed": false,
            "supportsSalt": true,
            "supportsOffset": false
        },
        {
            "algorithmType": "NGI_HMACSHA1",
            "supportsSeed": false,
            "supportsSalt": true,
            "supportsOffset": false
        }
    ]
}
```

6.2 Component Search

Request:

```
GET /gat/1.1/components?includeOS=false&includeContent=false
```

Response:


```
{
    "Components": [
        {
            "componentId": "ABC_123",
            "componentType": "NGI_software",
            "description": "The ABC123 Component",
            "size": 0,
            "algorithms": [
                {
                    "algorithmType": "NGI_SHA1",
                    "supportsSeed": false,
                    "supportsSalt": true,
                    "supportsOffset": false
                },
                {
                    "algorithmType": "NGI_HMACSHA1",
                    "supportsSeed": false,
                    "supportsSalt": true,
                    "supportsOffset": false
                }
            ]
        },
        {
            "componentId": "ABC_234",
            "componentType": "NGI_software",
            "description": "The ABC234 Component",
            "size": 0,
            "algorithms": [
                {
                    "algorithmType": "NGI_SHA1",
                    "supportsSeed": false,
                    "supportsSalt": true,
                    "supportsOffset": false
                },
                {
                    "algorithmType": "NGI_HMACSHA1",
                    "supportsSeed": false,
                    "supportsSalt": true,
                    "supportsOffset": false
                }
            ]
        }
    ]
}
```


Request:

```
GET /gat/1.1/components?endClientType=NGI_kiosk&endClientID=ABC_12345
```

```
    &includeOS=false&includeContent=false


Response:

{
    "components": [
        {
            "endClientType": "NGI_kiosk",
            "endClientID": "ABC_12345",
            "componentID": "ABC_123",
            "componentType": "NGI_software",
            "description": "The ABC123 Component",
            "size": 0,
            "algorithms": [
                {
                    "algorithmType": "NGI_SHA1",
                    "supportsSeed": false,
                    "supportsSalt": true,
                    "supportsOffset": false
                },
                {
                    "algorithmType": "NGI_HMACSHA1",
                    "supportsSeed": false,
                    "supportsSalt": true,
                    "supportsOffset": false
                }
            ]
        },
        {
            "endClientType": "NGI_kiosk",
            "endClientID": "ABC_12345",
            "componentID": "ABC_234",
            "componentType": "NGI_software",
            "description": "The ABC234 Component",
            "size": 0,
            "algorithms": [
                {
                    "algorithmType": "NGI_SHA1",
                    "supportsSeed": false,
                    "supportsSalt": true,
                    "supportsOffset": false
                },
                {
                    "algorithmType": "NGI_HMACSHA1",
                    "supportsSeed": false,
                    "supportsSalt": true,
                    "supportsOffset": false
                }
            ]
        }
    ]
}
```

6.3 Create Verification Task

Request:

```
POST
/gat/1.1/verificationTasks?componentId=ABC_123&algorithmType=NGI_SHA1
       &salt=2A3B4C5D6E7F8091A708192B3C4D5E6F
```

Response:

```
/gat/1.1/verificationTasks/345678
```

```
Request:

POST
/gat/1.1/verificationTasks?endClientType=NGI_kiosk&endClientID=ABC_12345
    &componentID=ABC_123&algorithmType=NGI_SHA1
    &salt=2A3B4C5D6E7F8091A708192B3C4D5E6F

Response:

/gat/1.1/verificationTasks/345678
```

6.4 Verification Task Details

Request:

```
GET /gat/1.1/verificationTasks/123456
```

Response:

```
{
    "verificationId": 123456,
    "componentId": "ABC_123",
    "verifyState": "NGI_complete",
    "algorithmType": "NGI_SHA1",
    "salt": "1A2B3C4D5E6F708192A3B4C5D6E7F809",
    "verifyResult": "ABC56DF718AA726FE9B72579BC45D1F2"
}
```

6.5 Verification Task Search

Request:

```
GET /gat/1.1/verificationTasks?componentId=ABC_123
```

Response:

```
{
    "VerificationTasks": [
        {
            "verificationId": 123456,
            "componentId": "ABC_123",
            "verifyState": "NGI_complete",
            "algorithmType": "NGI_SHA1",
            "salt": "1A2B3C4D5E6F708192A3B4C5D6E7F809",
            "verifyResult": "ABC56DF718AA726FE9B72579BC45D1F2"
        },
        {
            "verificationId": 234567,
            "componentId": "ABC_123",
            "verifyState": "NGI_complete",
            "algorithmType": "NGI_SHA1",
            "salt": "2A3B4C5D6E7F8091A2B3C4D5E6F70819",
            "verifyResult": "9B72579BC45D1F2ABC56DF718AA726FE"
        }
    ]
}
```

```
Request:

GET /gat/1.1/verificationTasks?endClientType=NGI_kiosk&endClientID=ABC_12345
    &componentID=ABC_123

Response:

{
    "verificationTasks": [
        {
            "verificationID": 123456,
            "endClientType": "NGI_kiosk",
            "endClientID": "ABC_12345",
            "componentID": "ABC_123",
            "verifyState": "NGI_complete",
            "algorithmType": "NGI_SHA1",
            "salt": "1A2B3C4D5E6F708192A3B4C5D6E7F809",
            "verifyResult": "ABC56DF718AA726FE9B72579BC45D1F2"
        },
        {
            "verificationID": 234567,
            "endClientType": "NGI_kiosk",
            "endClientID": "ABC_12345",
            "componentID": "ABC_123",
            "verifyState": "NGI_complete",
            "algorithmType": "NGI_SHA1",
            "salt": "2A3B4C5D6E7F8091A2B3C4D5E6F70819",
            "verifyResult": "9B72579BC45D1F2ABC56DF718AA726FE"
        }
    ]
```

```
}
```

6.6 Expected Results Detail

Request:

```
GET /gat/1.1/expectedResults?componentId=ABC_123&algorithmType=NGI_SHA1
    &supportsSalt=true
```

Response:

```
{
    "componentId": "ABC_123",
    "algorithmType": "NGI_SHA1",
    "expectedResultsList": [
        {
            "salt": "1A2B3C4D5E6F708192A3B4C5D6E7F809",
            "expectedResult": "ABC56DF718AA726FE9B72579BC45D1F2",
            "resultVersion": "1.1.0",
            "resultStatus": true,
            "expiration": "20140710T23:59:59.999-05:00"
        },
        {
            "salt": "2A3B4C5D6E7F8091A2B3C4D5E6F70819",
            "expectedResult": "9B72579BC45D1F2ABC56DF718AA726FE",
            "resultVersion": "1.1.0",
            "resultStatus": true,
            "expiration": "20140710T23:59:59.999-05:00"
        },
        {
            "salt": "2A3B4C5D6E7F8091A708192B3C4D5E6F",
            "expectedResult": "FE9B72579BC45D1F2ABC56DF718AA726",
            "resultVersion": "1.1.0",
            "resultStatus": true,
            "expiration": "20140710T23:59:59.999-05:00"
        }
    ]
}
```

END OF DOCUMENT

Released: 2019/07/11