Direct Connect Implementation Guide

API4133 | Document Version 4.3

Alvarado Manufacturing Company, Inc.
12660 Colony Street, Chino, CA 91710
Phone: +1 (909) 591-8431
Fax: +1 (909) 628-1403

Contents

Preface

Overview

This document provides an overview of Alvarado's Direct Connect interface. It defines API calls and provides examples of how Alvarado uses these items in GateLink for comparison purposes.

This document is intended to be used for reference purposes by a technical audience. It is not meant to instruct users how to develop an interface.

Support

For support, questions, or comments, please contact Alvarado's Integration Support department Monday – Friday, 8:00 AM – 4:00 PM Pacific time.

Phone: 909-591-8431

Email: integrationsupport@alvaradomfg.com

Direct Connect API

This information is for Direct Connect API version 1.3.0, release date 1/10/2020.

ValidateTicket

Normal online ticket validation request. When a device decodes a ticket, it sends a request to the server to determine whether the ticket is allowed to enter or not. The server determines whether the ticket is valid or invalid and sends that response to the device.

GetTicketInfo

Validation status request, i.e. was the ticket used, and if so, where? A request for ticket information functions like a test validation since it doesn't validate the ticket. Alvarado's ticket information request returns the following information.

- Ticket status (whether the ticket is valid or invalid)
- Validation information (shows device name, location, and validation time if the ticket was already validated)
- Other optional information (information like seat assignments or patron name, if available)

ForceValidation

Request to report an offline ticket validation. If Alvarado devices lose connectivity to the server (go offline), they use a local, simplified set of validation rules to determine whether a ticket should be admitted or not. After a device reestablishes connectivity to the server, it uploads all the tickets it admitted while offline. The server then retroactively determines whether the tickets that were admitted by the offline device were actually valid or invalid. The server identifies tickets validated this way as "forced valid" tickets.

VoidValidation

Request to undo/reverse a previously validated ticket. A ticket validation is voided if the validation unlocked a turnstile arms but there is no associated turnstile arm rotation (because the turnstile arms timed out and relocked). The server voids the ticket validation so that it can be scanned again and still be valid.

LoginAdministrator

This call functions differently depending on whether you are connecting to Alvarado TAS12 devices or handheld PGate devices.

For Alvarado TAS12 devices, this call requests access to a device's administrator functions menu. From this menu, users can perform manual ticket validations, perform device tests, change some device configuration settings, and exit the application. A numeric password is required to access the administrator functions menu. The call is used whenever a user enters a password and the device submits it to the server.

For Alvarado PGate devices, an operator login is optional and can be enabled or disabled in the device's configuration. When this functionality is enabled, operators must log into their devices before they can perform ticket validations. This call is used whenever a device operator submits a password/login from the SVT.

SynchronizeDevice

Device startup synchronization request – syncing server time and key device parameters. When the Direct Connect application starts on a device, the device requests the date/time and configuration settings from the server.

Files Updated – TAS12 Direct Connect Devices

When TAS12 Direct Connect devices receive updates from the server, they update several configuration text files in the device's *FlashDisk\TAS* folder.

- Messages.txt – File contains a list of response codes and the user-friendly text responses that accompany them. The text responses display when a ticket is validated while viewing the *Ticket Validation* screen from the administrator functions menu.
- Operators.txt – File contains a list of administrator logins that can be used to access the administrator functions menu.
- PFormat.txt – File contains a print format used if the device is equipped with a printer.
- TAS.txt – The server updates the fields containing validation masks, which are used to determine whether a ticket is valid or invalid if the device is offline.

Files Updated – PocketGate Direct Connect Devices

When PocketGate Direct Connect devices receive updates from the server, they are saved locally to the device. These files are not directly editable. The devices receive updates for the following configuration information.

- Messages to be displayed when the server returns a response code after a validation request
- Operator logins that are used to access the PocketGate *Home* screen, if applicable
- Print formats that are used to determine how seat locator slips are printed
- Validation masks that are used to determine whether a ticket is valid or invalid if the device is offline.

Ping

During normal online operation, devices periodically send a "Ping" request to the host system, to ensure that there is network connectivity. Acknowledgement of the ping request by the host system's Web service is required for the device to continue operating in online mode. If a ping request is not acknowledged by the host, the device will switch to offline validation mode and continue to operate in this mode until a ping request is acknowledged by the host.

While operating in online mode, ping requests are sent from the device approximately every 40 seconds. While in offline mode, ping requests are sent every few seconds to bring the device back online as quickly as possible.

RESTful JSON Interface

Direct Connect devices are configured by default to use a RESTful JSON interface.

IMPORTANT

- Direct Connect devices using JSON use the Pascal Case naming convention for all fields. For example: IpAddress, SiteName.
- The endpoints use the **POST http method**.

- Minimum TAS DC version: 1.3.1
- Minimum PocketGateDC version: 1.3.0
- Minimum ValiD8 version: 1.0.2

Refer to the user documentation for the relevant device for configuration information.

Default Endpoint Map

Call	Endpoint
ValidateTicket	/validate
GetTicketInfo	/info
ForceValidation	/force
VoidValidation	/void
LoginAdministrator	/login
SynchronizeDevice	/sync
Ping	/ping

- The *ValidateTicket* call does not require the endpoint to be called "/validate". However, whatever name you create for it must follow the structure outlined in the **API Call Structure** section on page 18.
- The *Ping* and *SynchronizeDevice* calls are required for Direct Connect devices to go online.

Sandbox

Alvarado provides a sandbox for development and testing. You can access the sandbox at https://sandbox.alvaradomfg.com/directconnect.

Refer to device documentation on how to configure the server. Use the authentication header in the examples in the next section to connect to the sandbox.

JSON Examples

Authentication

Direct Connect devices use the Basic Authentication method, passing requests in the header. The server returns a response body in JSON format with status code 200 (OK).

An example of an authorization header in a JSON request:

```
Header Name: Authorization
Header Value: Basic YWRtaW4vYWx2YXJhZG8=
```

Badly formatted JSON POST requests or requests that cannot be executed return an error message and status code 400 (bad request). A failed authorization generates an error response with status code 401 (unauthorized).

```
{
	“Status”: “Error”,
	“Message”: “Authorization Required”
}
```

Device Synchronization

Sample request body (sent from device number 30 with IP address 192.168.0.30).

```
{
	"IpAddress": "192.168.0.30",
	"DeviceNo": 30,
	"SiteName": "Wild Arena",
	"LoginCode": "admin",
	"DateTime": "2019-12-28T14:12:16"
}
```

Sample JSON response:


```
{
	"DateTime":"2019-12-28T14:12:18Z",
	"DeviceNo":30,
	"SiteName":"Alvarado Test",
	"DeviceDescription":"TAS 30",
	"GreenTimeout":10, "redTimeout":3,
	"PrintFormat":"^XA\n^LL500\n^FO50,10^XGE:<$LOGO>^FS\n^FO50,100^ADN,32,12^F
	D<$EVENT>^FS\n^FO50,150^ADN,18,10^FD<$LINE1>^FS\n^FO50,180^ADN,18,10^FD<$L
	INE2>^FS\n^FO50,210^ADN,18,10^FD<$LINE3>^FS\n^FO5<$DATE>
	<$TIME>^FS\n^FO50,300^ADN,32,16^FDSect:<$SECTION> Row:<$ROW> Seat:
	<$SEAT>^FS\n^FO80,350^BCN,60,Y,N,N^FD<$BARCODE>^FS\n^FO50,460^A0,20,16^FD(
	c)2019Alvarado Mfg. Co., Inc.^FS\n^CN1\n^PN0\n^XZ\n",
	"ResponseCodes":
	[
		{"Code":0,"description":"WEB Valid"},
		{"Code":10,"description":"WEB Reentry"},
		{"Code":11,"description":"WEB Invalid Event"},
		{"Code":12,"description":"WEB Invalid Date"},
		{"Code":13,"description":"WEB Invalid Time"},
		{"Code":14,"description":"WEB Invalid Day"},
		{"Code":15,"description":"WEB No Access"},
		{"Code":16,"description":"WEB Invalid Location"},
		{"Code":21,"description":"WEB Used"},
		{"Code":31,"description":"Ticket Cancelled"},
		{"Code":32,"description":"Ticket Lost"},
		{"Code":33,"description":"Ticket Stolen"},
		{"Code":41,"description":"Already In"},
		{"Code":42,"description":"Already Out"},
		{"Code":43,"description":"Exit without Enter"},
		{"Code":51,"description":"Not Found"},
		{"Code":52,"description":"Rescan"},
		{"Code":53,"description":"Invalid Mask"}
	],
	"ValidationMasks":
	[
		{"Mask":"001??????????"},
		{"Mask":"002??????????"},
		{"Mask":"003??????????"}
	]
}
```


Administrator Login

Sample request body (sent from device number 30 with IP address 192.168.0.30).


```
{
	"IpAddress":"192.168.0.30",
	"DeviceNo":30,
	"DateTime":"2019-12-28T14:12:52",
	"SiteName":"Test Server",
	"LoginCode":"123"
}
```

Sample JSON response for a valid login (*result* field with Boolean value of “true”).

```
{
	"LoginCode":"123",
	"Result":true
}
```

Sample JSON response for an invalid login (*result* field with Boolean value of “false”).

```
{
	"LoginCode":"123",
	"Result":false
}
```

Ticket Validation

Sample request body (sent from device number 30 with IP address 192.168.0.30).

```
{
	"IpAddress":"192.168.0.30",
	"DeviceNo":30,
	"DateTime":"2019-12-28T14:13:59",
	"Direction":"+",
	"Ticket":"12374377416323",
	"SiteName":"Test Server",
	"LoginCode":"123"
}
```

Sample JSON response for valid ticket "12374377416323". The *ticket* and *responseCode* fields are required.

```
{
	"Ticket":"12374377416323",
	"ResponseCode":0,
	"PrintSls":false,
	"EventDescription":"",
	"EventTime":"",
	"PatronName":"Chris Brown",
	"Level":"2",
	"Section":"3",
	"Row":"7",
	"Seat":"43",
	"Line1":"",
	"Line2":"",
	"Line3":"",
	"Line4":"",
	"Line5":""
}
```

Sample JSON response for invalid ticket "32020879557390".

```
{
	"Ticket":"32020879557390",
	"ResponseCode":51,
	"PrintSls":false,
	"EventDescription":"",
	"EventTime":"",
	"PatronName":"",
	"Level":"",
	"Section":"",
	"Row":"",
	"Seat":""
}
```

Ticket Info

Sample request body (sent from device number 30 for ticket “12304329838356”).

```
{
	"IpAddress":"192.168.0.31",
	"DeviceNo":30,
	"DateTime":"2019-12-28T15:12:21",
	"Direction":"+",
	"Ticket":"12304329838356",
	"SiteName":"Test Server",
	"LoginCode":"123"
}
```

Sample JSON response for valid and used ticket "12304329838356". The *ticket* and *responseCode* fields are required.

```
{
	"Ticket":"12304329838356",
	"ResponseCode":0,
	"PatronName":"Chris Brown",
	"Level":"2",
	"Section":"3",
	"Row":"0",
	"Seat":"43",
	"UsedDateTime":"2019-12-28T15:02:21Z",
	"UsedLocation":"Main Entrance"
}
```

Void Ticket Validation

Sample request body (sent from device number 30).

```
{
	"IpAddress":"192.168.0.30",
	"DeviceNo":30,
	"DateTime":"2019-12-28T14:37:58",
	"Direction":"+",
	"Ticket":"12304362283098",
	"SiteName":"Test Server",
	"LoginCode":""
}
```

Sample JSON response for void ticket “123043362283098”. The response requires the *status* field to be “OK”. The *message* field is optional and can be used for logging or debugging purposes.

```
{
	"Status":"OK",
	"Message":"Ticket 12304362283098 voided"
}
```

Forced Offline Ticket Validation

Sample request body for a forced entry validation (sent from device number 30).

```
{
	"IpAddress":"192.168.0.30",
	"DeviceNo":30,
	"DateTime":"2019-12-28T15:09:23",
	"Direction":"+",
	"Ticket":"12304370938252",
	"SiteName":"Test Server",
	"LoginCode":"123"
}
```

Sample JSON response for offline forced ticket “12304370938252”. The response requires the *status* field to be “OK”. The *message* field is optional and can be used for logging or debugging purposes.

```
{
	"Status":"OK",
	"Message":"Ticket 12304370938252 forced"
}
```

Ping

Sample request body (sent from device number 30 with IP address 192.168.0.30).

```
{
	"IpAddress": "192.168.0.30",
	"DeviceNo": 102,
	"SiteName": "Wild Arena",
	"LoginCode": "admin"
}
```

Sample response. The *Status* field must return “OK”. (Note: The *Message* field below is optional and can be used for logging or debugging.)

```
{
	"Status": "OK",
	"Message": "Ping from 30"
}
```

SOAP XML Interface

WSDL and Sample Web Services

Alvarado provides each customer with the WSDL for Direct Connect. If this was not provided to you, contact Alvarado's Integration Support department.

Alvarado can provide sample web service starter projects in C# and Java upon request for you to use for reference purposes, if required.

Alvarado provides a sandbox for development and testing. You can access the sandbox at https://sandbox.alvaradomfg.com/soapdirectconnect/TasDirectConnectService.exe/soap/ITasDirectConnect. Refer to device documentation for how to configure the server.

Alvarado Example Responses

This section provides examples of how Alvarado handles Response Codes, images, and LED settings. You can find more details about these items in the *Direct Connect Advanced Configuration Guide*.

Response Codes

Each time a Ticket is presented to an Admission Device, a Response Code is returned. The Response Codes returned depends on whether the Ticket is valid or invalid, Ticket and Event configurations, and scanning circumstances.

Direct Connect uses response codes 0–10 for valid tickets and 11–99 for invalid tickets.

You can assign any valid or invalid meaning you want to the different response codes, with a few exceptions. Some response codes have hard-coded offline behavior. You can change the name or message these codes return, but not their functionality.

Hard-Coded Offline Behavior

Response Code	Response Code Behavior
21	Online: Ticket was already used, according to the server. Offline: Ticket was already validated by that device when it was offline.
55	Online: N/A Offline: The scanned ticket did not match an offline mask.
59	Online: N/A Offline: Unknown error
72	Online/Offline: The scanned ticket's barcode length did not meet the minimum threshold.
73	Online/Offline: The scanned ticket's barcode length did not meet the maximum threshold.

Sample Alvarado Response Codes

The table below shows the different response codes available to Alvarado's GateLink software during ticket validation. The table lists the response code number, the name it has been assigned for user-friendliness, and a description of when that response code would be returned.

This is provided for **example purposes only**. You are responsible for defining what each valid and invalid response code means in your interface.

Response Code	Response Code Name	Response Code Description
0	Good Ticket	A Ticket with Control Number 0 was validated. This is the default Control Number for valid Tickets.
1	Good 1	A Ticket with Control Number 1 was validated.
2	Good 2	A Ticket with Control Number 2 was validated.
3	Good 3	A Ticket with Control Number 3 was validated.
4	Good 4	A Ticket with Control Number 4 was validated.
5	Good 5	A Ticket with Control Number 5 was validated.
6	Good 6	A Ticket with Control Number 6 was validated.
7	Good 7	A Ticket with Control Number 7 was validated.
8	Good 8	A Ticket with Control Number 8 was validated.
9	Good 9	A Ticket with Control Number 9 was validated.
10	Reentry	The Ticket was denied entry because it was already used for entry into a Facility, but the Device Operator used the Reentry command to allow the Patron to enter.
11	Invalid Event	The Ticket was denied entry because the Event to which the Ticket was assigned was not valid when the Ticket was scanned. For example, an Event is valid only on Wednesday and the Ticket was scanned on Tuesday.
12	Invalid Date	The Ticket was denied entry because it was scanned on the wrong day for an Event that is valid across multiple days. For example, an Event is valid from Monday to Friday, but a Ticket is only valid Monday through Wednesday. If the Ticket is scanned on Thursday it will receive this Response Code.
13	Invalid Time	The Ticket was denied entry because it was scanned at the wrong time during a valid Event. For example, an Event is valid from 3:00 pm to 10:00 pm, but the Ticket is scanned at 1:00 pm.
14	Invalid Day	The Ticket was denied entry because it was scanned on a day of the week the Event was not valid. For example, the Event was set to be valid **SMtwtFS** (Valid on Sunday, Monday, Friday, and Saturday. Invalid on Tuesday, Wednesday, and Thursday.) in Field 13 of the Event (.evt) File. If a Ticket is scanned on Tuesday, Wednesday, or Thursday, it will receive this Response Code.
15	Invalid Access	The Ticket was denied entry because it was scanned at an incorrect Gate Map location or had an incorrect Access Level.
16	Invalid Location	The Ticket was denied entry because it was scanned at an incorrect Scan Location or Secondary Scan Location.
21	Used in Event	The Ticket was denied entry because it was already validated for the Event.
22	Used in Day	The Ticket was denied entry because it was already validated for the day during a multiple-day Event.
23	Used in Hour	The Ticket was denied entry because it was already validated for the hour during an Event with an Hourly Limit.
24	Used Days	The Ticket was denied entry because it exceeded the Maximum Days limit set by the Event.
26	Used at Second Location	The Ticket was scanned at a Secondary Location, but was denied entry because it was already scanned and admitted at a Secondary Scan Location earlier in the Event.
27	No Entries Remaining	A Group Ticket was denied entry because there were no Product Type 1 or Product Type 2 entries remaining.
31	Cancelled 1	The Ticket was denied entry because it was cancelled using Cancel Field 1.
32	Cancelled 2	The Ticket was denied entry because it was cancelled using Cancel Field 2.
33	Cancelled 3	The Ticket was denied entry because it was cancelled using Cancel Field 3.
34	Cancelled 4	The Ticket was denied entry because it was cancelled using Cancel Field 4.
35	Cancelled 5	The Ticket was denied entry because it was cancelled using Cancel Field 5.

Response Code	Response Code Name	Response Code Description
36	Cancelled 6	The Ticket was denied entry because it was cancelled using Cancel Field 6.
37	Cancelled 7	The Ticket was denied entry because it was cancelled using Cancel Field 7.
38	Cancelled 8	The Ticket was denied entry because it was cancelled using Cancel Field 8.
41	Already In	The Ticket was denied entry because it was presented for reentry to an Entry / Exit Event without having been scanned out.
42	Already Out	The Ticket was denied exit because it was presented to be scanned out during an Entry / Exit Event without having reentered.
43	Exit without Enter	The Ticket was denied entry because it was presented for exit during an Entry / Exit Event without first being scanned in.
46	Invalid Ticket Price	The Ticket was denied entry because it did not fall within the Ticket Price Validation Range.
47	Invalid User Code 1	The Ticket was denied entry because it did not match the User Defined 1 mask.
48	Invalid User Code 2	The Ticket was denied entry because it did not match the User Defined 2 mask.
49	Invalid User Code 3	The Ticket was denied entry because it did not match the User Defined 3 mask.
51	Not Found	The Ticket was not found in the database.
52	Rescan	The Ticket was scanned again less than 10 seconds after being validated.
53	External Validation Failure	GateLink could not communicate with the External Validation Service when this Ticket was presented to an Admission Device.
54	Invalid Checksum	
59	Other	GateLink could not determine whether the Ticket was valid or invalid because data from the database could not be read or was corrupt.
99	Valid Exit	The Ticket was scanned out successfully during an Entry / Exit Event. **NOTE:** Response 99 is **invalid** in Direct Connect applications.

Image Files

TAS12 Direct Connect devices use bitmap images and ValiD8 Direct Connect devices use PNG images to display instructions and validation information to patrons. The devices come with four images by default.

- Go – Displays when any valid ticket is scanned.
- Stop – Displays when any invalid ticket is scanned.
- Scan – Idle screen for when the device is waiting for a ticket scan.
- Wait – Displays when the device is launching the application or processing information.

Each individual valid or invalid response code can have an associated image. The image would display whenever the server returns that response code instead of the generic “Go” or “Stop” image.

For more information about image requirements, see the advanced configuration guide for your product. Contact Alvarado Entertainment Support for assistance.

LED Settings

Most TAS12 devices have red, yellow, and green LEDs on the back of the head that show device operators whether the ticket is valid or invalid. The LEDs can have different illumination settings based on which valid response code (0-10) the server returns.

For more information about image requirements, see the advanced configuration guide for your product. Contact Alvarado Entertainment Support for assistance.

API Call Structure

NOTE	JSON endpoints use the **POST http method**.

ValidateTicket

Client Request Structure

The parameters for this call are shown below.

Parameter	Description	Data Type	Details/Format
IpAddress	Device IP address	String	IPv4 (for example: 192.168.0.1)
DeviceNo	Device number assigned to the device	Integer	1-999
DateTime	Date/time stamp when the device validated the ticket.	String	YYYY-MM-DDThh:mm:ss
Direction	Indicates whether the ticket was scanned at an entry or exit device.	String	"Plus symbol" (+) for scans at entry devices. "Minus symbol" (-) for scans at exit devices.
Ticket	Barcode or identifier that will be validated	String	Alphanumeric
SiteName	Identifies where device is located	String	Alphanumeric
LoginCode	PGate devices only: Identifies the logged-in operator on each request that requires a login. Assumes the device is in Operator Mode, where a login is required before validations can be performed. TAS devices only: Identifies the logged-in administrator when they perform a manual validation.	String	Alphanumeric NOTE: Alvarado TAS12 and SVT devices are only capable of submitting numeric login codes.

ValidateTicket - Structure of Host Response for VALID Tickets

Parameter	Description	Data Type	Details/Format	Required?
Ticket	Barcode or identifier that will be validated	String	Alphanumeric	YES
ResponseCode	Numeric code that differentiates between different types of valid or invalid tickets.	Integer	0-10 for valid response codes. 11-99 for invalid response codes.	YES
PrintSls	Asks the validation service whether the device should print a seat locator slip or not.	String	"Yes": Print a seat locator slip. "No": Do not print a seat locator slip.	NO (Defaults to FALSE if field is blank)
EventDescription	User-friendly description given to the event.	String	Alphanumeric	NO
EventTime	Start-time defined for the event.	String	Alphanumeric	NO
PatronName	First and last name of the patron to which the ticket is assigned. Can be printed on tickets or receipts printed at admission devices.	String	Alphanumeric	NO
Level	Used to indicate seating level in assigned-seating events.	String	Alphanumeric	NO
Section	Used to indicate seating section in assigned-seating events.	String	Alphanumeric	NO
Row	Used to indicate seating row in assigned-seating events.	String	Alphanumeric	NO
Seat	Used to indicate seat number in assigned-seating events.	String	Alphanumeric	NO
Line1	Available fields that can be used to print data on tickets or receipts printed at admission devices.	String	Alphanumeric	NO
Line2				
Line3				
Line4				
Line5				

GetTicketInfo

Client Request Structure

The parameters for this call are shown below.

Parameter	Description	Data Type	Details/Format
IPAddress	Device IP address	String	IPv4 (for example: 192.168.0.1)
DeviceNo	Device number assigned to the device	Integer	1-999
Ticket	Scan timestamp	String	YYYY-MM-DDThh:mm:ss
DateTime	This field is not used in this request.	String	"Plus symbol" (+) for scans at entry devices. "Minus symbol" (-) for scans at exit devices.
Direction	This field is not used in this request.	String	Alphanumeric
SiteName	Identifies where device is located	String	Alphanumeric
LoginCode	PGate: Identifies the logged-in operator on each request that requires a login. TAS: This field is not used in this request.	String	Alphanumeric NOTE: Alvarado TAS12 and SVT devices are only capable of submitting numeric login codes.

GetTicketInfo - Structure of Host Response

Parameter	Description	Data Type	Details/Format	Required
Ticket	Barcode or identifier that will be validated	String	Alphanumeric	YES
ResponseCode	Numeric code that differentiates between different types of valid or invalid tickets.	Integer	0-10 for valid response codes. 11-99 for invalid response codes.	YES
UsedDateTime	Timestamp indicating the most recent date/time the ticket was validated.	String	YYYY-MM-DDThh:mm:ss (Null if Not Used)	YES (If ticket has been previously validated)
UsedLocation	The device name or description of the device that performed the most recent validation.	String	Alphanumeric (Null if Not Used)	YES (If ticket has been previously validated)
PatronName	First and last name of the patron to which the ticket is assigned. Can be printed on tickets or receipts printed at admission devices.	String	Alphanumeric	NO
Level	Used to indicate seating level in assigned-seating events.	String	Alphanumeric	NO
Section	Used to indicate seating section in assigned-seating events.	String	Alphanumeric	NO
Row	Used to indicate seating row in assigned-seating events.	String	Alphanumeric	NO
Seat	Used to indicate seat number in assigned-seating events.	String	Alphanumeric	NO

ForceValidation

This call sends a synchronization request to report an offline validation to the host system.

The SOAP XML interface must respond with an **HTTP 1.1/200 OK** response and a self-closing *ForceValidation* tag.

For calls that don't show a response table, the RESTful JSON interface requires a `"status": "OK"` in the response.

Client Request Structure

The parameters for this call are shown below.

Parameter	Description	Data Type	Details/Format
IPAddress	Device IP address	String	IPv4 (for example: 192.168.0.1)
DeviceNo	Device number assigned to the device.	Integer	1-999
DateTime	Date/time stamp when the device validated the ticket.	String	YYYY-MM-DDThh:mm:ss
Direction	Indicates whether the ticket was scanned at an entry or exit device.	String	"Plus symbol" (+) for scans at entry devices. "Minus symbol" (-) for scans at exit devices.
Ticket	Barcode or identifier that was validated offline by the device.	String	Alphanumeric
SiteName	Identifies where device is located	Alphanumeric	Alphanumeric
LoginCode	PGate: Identifies the logged-in operator on each request that requires a login. TAS: Identifies the logged-in administrator when they perform a manual validation.	Alphanumeric	Alphanumeric NOTE: Alvarado TAS12 and SVT devices are only capable of submitting numeric login codes.

VoidValidation

This call sends an online request to void or reverse a previous validation.

The SOAP XML interface must respond with an **HTTP 1.1/200 OK** response and a self-closing *ForceValidation* tag.

For calls that don't show a response table, the RESTful JSON interface requires a `"status": "OK"` in the response.

Client Request Structure

The parameters for this call are shown below.

Parameter	Description	Data Type	Details/Format
IPAddress	Device IP address	String	IPv4 (for example: 192.168.0.1)
DeviceNo	Device number assigned to the device	Integer	1-999
DateTime	Date/time stamp when the ticket validation occurred	String	YYYY-MM-DDThh:mm:ss
Ticket	Barcode or identifier of the ticket that was voided	String	Alphanumeric

Server Response/Acknowledgement for VoidValidation Request

The SOAP XML interface must respond with an **HTTP 1.1/200 OK** response and a self-closing *ForceValidation* tag.

For calls that don't show a response table, the RESTful JSON interface requires a "status": "OK" in the response.

LoginAdministrator

This call sends a request to gain access to the device's administrator menu (TAS12 devices) or to allow the operator to perform ticket validations (SVT devices).

Client Request Structure

The parameters for this call are shown below.

Parameter	Description	Data Type	Details/Format
IPAddress	Device IP address	String	IPv4 (for example: 192.168.0.1)
DeviceNo	Device number assigned to the device	Integer	1-999
DateTime	Date/time stamp when login attempt occurred	String	YYYY-MM-DDThh:mm:ss
SiteName	Identifies where device is located	String	Alphanumeric
LoginCode	PGate: Identifies the logged-in operator on each request that requires a login. TAS: Requests access to a device's administrator functions menu	String	Alphanumeric NOTE: Alvarado TAS12 and SVT devices are only capable of submitting numeric login codes.

LoginAdministrator - Structure of Host Response

Parameter	Description	Data Type	Details/Format	Required
LoginCode	PGate: Identifies the logged-in operator on each request that requires a login. TAS: Identifies the logged-in administrator when they perform a manual validation.	String	Alphanumeric NOTE: Alvarado TAS12 and SVT devices are only capable of submitting numeric login codes.	YES
Response	Was the login successful?	String	YES NO	YES

SynchronizeDevice

This call sends a request to sync with the server time and other key operating parameters.

Client Request Structure

Parameter	Description	Data Type	Details/Format
IPAddress	Device IP address	String	IPv4 (for example: 192.168.0.1)
DeviceNo	Device number assigned to the device	Integer	1-999
DateTime	Date/time stamp when the synchronization request occurred	String	YYYY-MM-DDThh:mm:ss
SiteName	Identifies where device is located	String	Alphanumeric
LoginCode	PGate: Identifies the logged-in operator on each request that requires a login. TAS: Not used.	String	Alphanumeric NOTE: Alvarado TAS12 and SVT devices are only capable of submitting numeric login codes.

SynchronizeDevice - Structure of Host Response

Parameter	Description	Data Type	Details/Format	Required
DateTime	Current date and time on the host server	String	YYYY-MM-DDThh:mm:ss	YES
DeviceNo	Device number assigned to the device receiving the synchronization request. This should only be returned if the device number needs to be assigned or updated. Otherwise, the parameter should be empty.	Integer	1-999	*NO
DeviceDescription	Description of the device that the is receiving the synchronization request. The API service sets this when it synchronizes.	String	Alphanumeric	*NO
****GreenTimeout**	Time (in seconds) the green LED remains illuminated and/or the valid response remains displayed before returning to scan mode.	Integer	1-60	*NO
****RedTimeout**	Time (in seconds) the red LED remains illuminated and/or the invalid response remains displayed before returning to scan mode.	Integer	1-60	*NO
PrintFormat	Format for printing seat locator slips or validation receipts from the device, if equipped with a printer	String	Alphanumeric – ZPL Code (Zebra Printer Language)	*NO

Code/ Description **(returned as ResponseCodes)**	Numeric code that differentiates between different types of valid or invalid tickets.	Integer	0-10 for valid response codes. 11-99 for invalid response codes. Any **ResponseCodeItem** in response will overwrite locally cached description for associated code.	*NO
Mask **(returned as ValidationMasks)**	List of up to eight validation masks that can be stored locally on the device. Masks are used for validating tickets when the device is not connected to the validation server. example: **1*2?**	String	Pattern of characters with "?" serving as wildcard characters for variable single character and "*" for matching one or more characters. Any **ValidationMaskItem** in response will overwrite locally cached masks for the associated mask number. For example, if two masks are included in response, mask 1 and 2 will be overwritten. If three masks are included, mask 1, 2 and 3 are overwritten, etc., up to a total of eight (8) masks.	*NO

***If parameter is not included in response, then locally cached parameters will persist.**

**** Not applicable to PGate Direct Connect devices.**

Ping

This call is periodically sent to check for host system connectivity.

The SOAP XML interface must respond with an **HTTP 1.1/200 OK** response and a self-closing *ForceValidation* tag.

For calls that don't show a response table, the RESTful JSON interface requires a `"status": "OK"` in the response.

Client Request Structure

Parameter	Description	Data Type	Details/Format
IPAddress	Device IP address	String	IPv4 (for example: 192.168.0.1)
DeviceNo	Device number assigned to the device	Integer	1-999
SiteName	Not used in this request	String	Alphanumeric
LoginCode	Not used in this request	String	Alphanumeric

Revision History

Revision	Creation Date	Author	Revision History/Description
1.0	8/23/2016	T Laib	Original Document
1.1	8/26/2016	T Laib	Corrections and clarifications
2.0	9/14/2017	S Yang, D Bohannon	Updated formatting. Some minor content changes
3.0	5/6/2019	D Bohannon	Updated content.
3.1	12/12/2019	D Bohannon	Minor edit.
4.0	1/16/2020	D Bohannon	Added JSON information.
4.1	3/10/2020	D Bohannon	Minor edits.
4.2	3/19/2020	D Bohannon	Minor edits.
4.3	5/10/2021	D Bohannon	Minor edits.