



TECHNICAL NOTES

Player Provisioning and Recovery

INTRODUCTION

When developing for a large, distributed network of BrightSign players, it is important to consider how to tackle logistical issues that are part of such deployments. This technical note will outline how to develop a robust implementation that can meet the following requirements for an effective distributed player network:

1. Each player can provision itself an autorun when deployed in the field.
2. Once provisioned, each player should be able to communicate with a "request handler" server to periodically update its autorun.
3. If a player enters an error state that cannot be fixed by rebooting (resulting from an error caused by the autorun script, a bug in the firmware, or filesystem corruption caused by external power loss or card failure), the player and server can initiate a recovery process that will allow the player to return to working order.

To meet the above requirements, two separate but complimentary systems must be developed:

- A set of *autorun* files (*.brs* or *.bas*) that can play content, facilitate fault recovery, or provision the player for an HTML/JavaScript playback environment
- A *request handler* server, which processes HTTP GET requests sent by players. The request handler delivers an autorun file or other HTTP response depending on device information contained within the header of the request. The request handler database can also contain conditional flags for each player, allowing administrators to push new autorun scripts onto specified players in the field.

Provisioning and Recovery Process Overview

The following sections outline the different stages of the provisioning and recovery process as executed by the player: initial steps, override recovery, periodic recovery, and last-resort recovery. These descriptions include only information that is pertinent to the provisioning and recovery processes:

Initial Steps

The following steps occur every time the BrightSign player boots up:

1. The player determines if it has a network connection (Ethernet and/or WiFi). This step occurs before the player attempts to obtain an IP address.
2. The player checks the registry to determine if it is configured for networking.
3. The player checks the registry to determine if script debugging is enabled or disabled. If script debugging is enabled, the provisioning or recovery process will not take place. Script debugging is disabled by default.
4. The player carries out the DHCP process to obtain an IP address. If configured for Option 43, the DHCP server should provide a new provisioning/recovery URL to the player (see the **Recovery and Bootstrapping with DHCP Option 43** technical note for more details). Otherwise, the player will use the provisioning/recovery URL value already stored in the registry; this value is blank by default.

Override Recovery

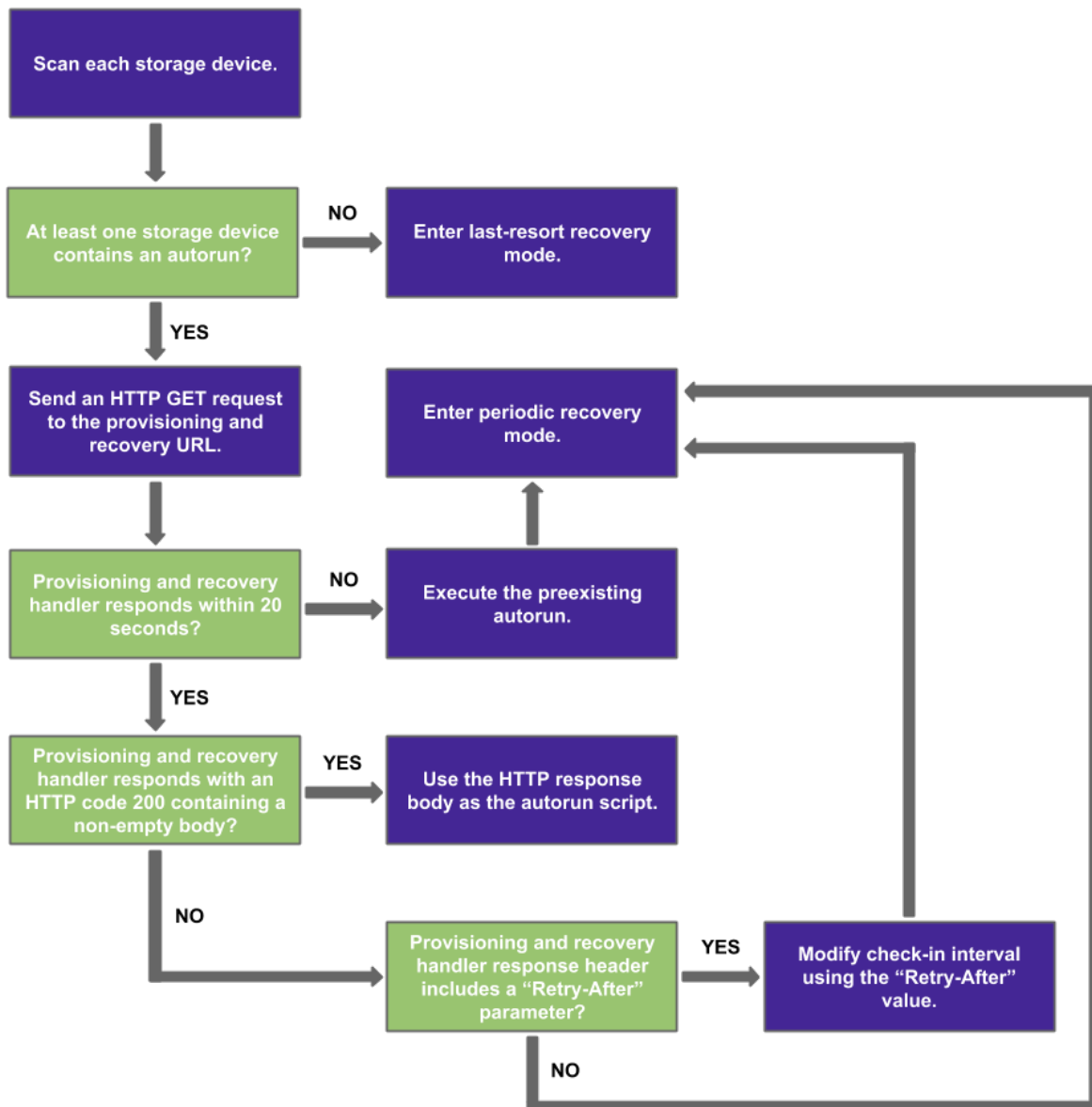
Once the boot process has been completed, the player scans each storage device to determine if one or more devices contain an autorun file:

1. If at least one of the storage devices contains an autorun file, the player will delay executing the autorun while a background process sends an HTTP GET request (containing parameters describing the player in the header) to the provisioning/recovery URL:
 - a. If the request handler responds using an HTTP code 200 with a non-empty body, the player will use the response body as the new autorun script.

- b. If the request handler responds with an HTTP code 204, or an HTTP code 200 containing an empty message body, the player will execute the preexisting autorun script as normal. The response may also include a "Retry-After" header indicating (in seconds) how frequently the player should check in with the request handler. If this parameter is not specified, the check-in interval will default to 2 hours.
- c. If no response is received within 20 seconds, the preexisting autorun will be executed. The player will send another request to the request handler after 5 minutes, then after 10 minutes. It will then make a request once every 2 hours. If multiple storage devices contain an autorun file, the player will select the first autorun in the following order: USB, SD, SD2, SSD.

Note: *If it takes too long to check the integrity of a storage device, then that device might be skipped in the order of autorun selection. For this reason, we do not recommend building a system that relies on the stability of this order.*

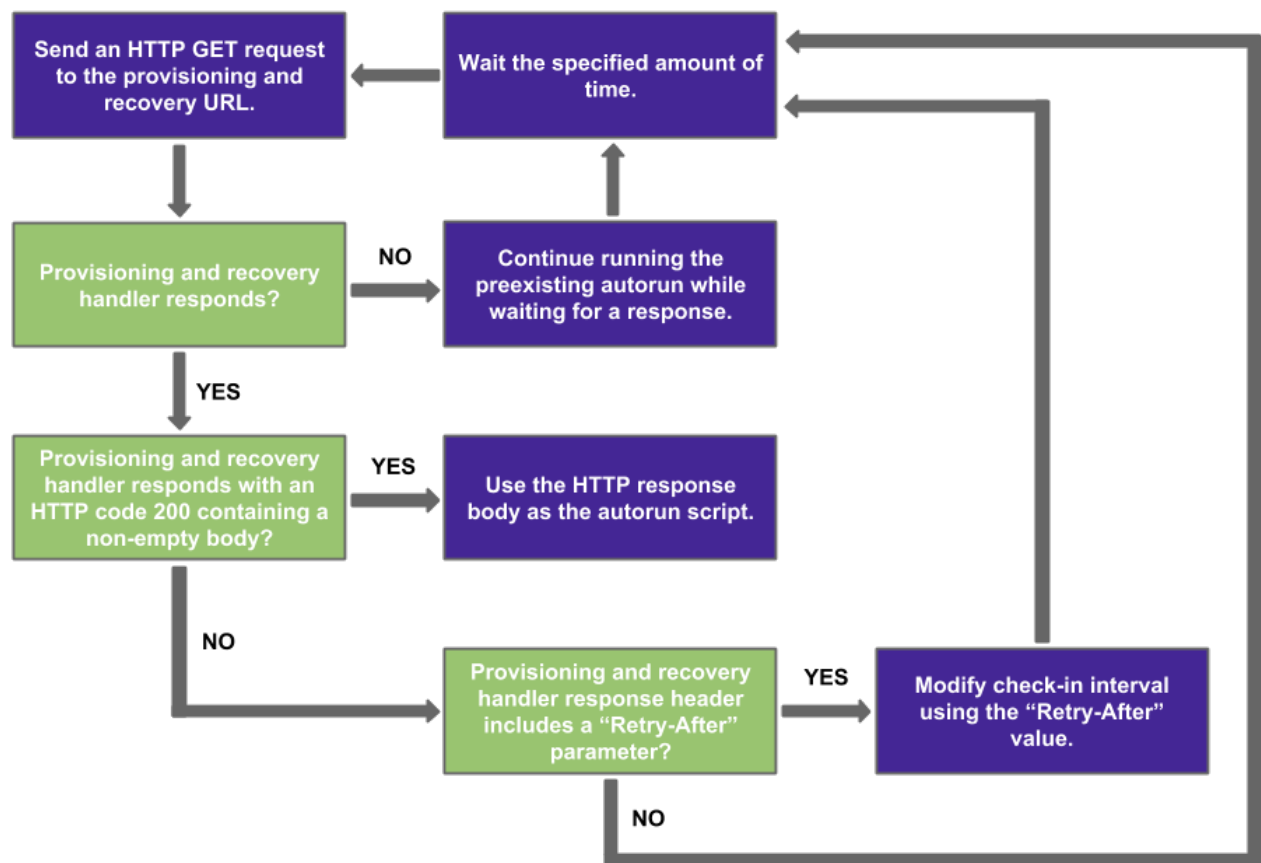
2. If none of the storage devices contain an autorun, the player will enter last-resort recovery mode.



Periodic Recovery

The player will periodically send an HTTP GET request to the provisioning and recovery URL to determine if the device should be updated.

1. If no response is received, the preexisting autorun will continue without interruption.
2. If the request handler responds with an HTTP code 204, or an HTTP code 200 containing an empty message body, the preexisting autorun will continue without interruption. The response may also include a "Retry-After" header indicating (in seconds) how frequently the player should check in with the request handler. If this parameter is not specified, the check-in interval will default to 2 hours.
3. If the request handler responds using an HTTP code 200 with a non-empty body, the player will use the response body as the new autorun script.



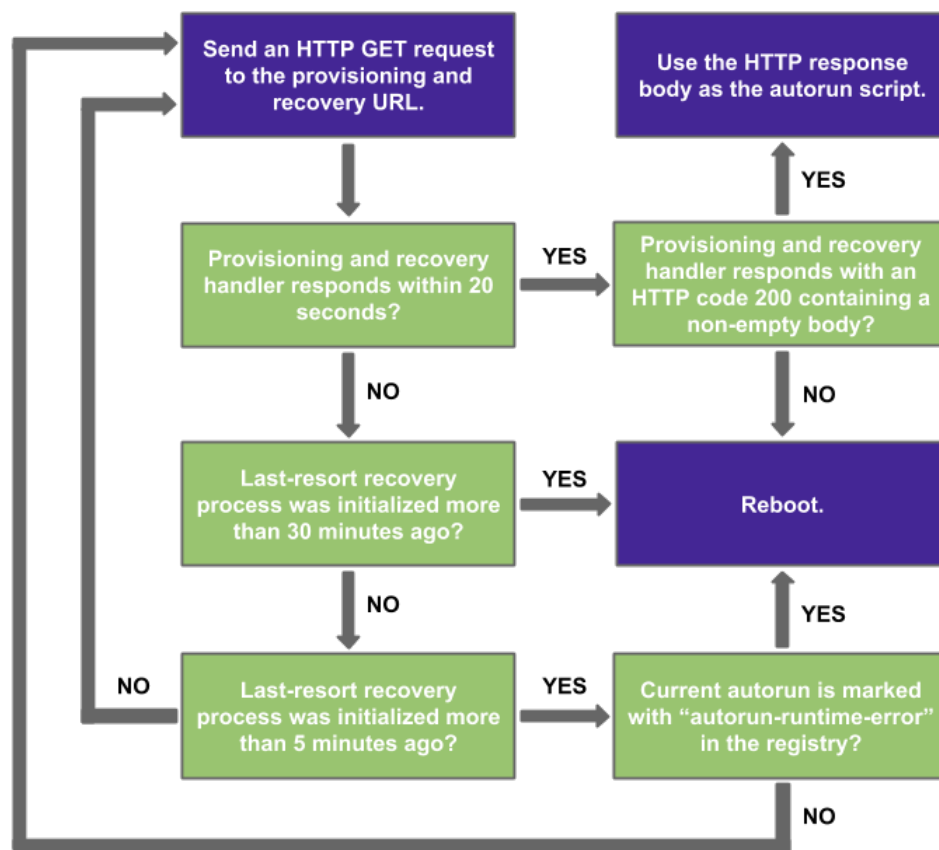
Last-Resort Recovery

If none of the attached storage devices contain an autorun file, or if the designated autorun encounters an `autorun-load-error` or `autorun-runtime-error`, then last-resort recovery will begin.

1. The player sends an HTTP GET request to the provisioning/recovery URL. The header of this request contains information about the player, as well as the status of the storage devices and

autorun files on the player: This allows the request handler to determine the correct course of action based on the attributes and current status of the player.

- If the request handler responds using an HTTP code 200 with a non-empty body, the player will use the response body as the new autorun script. It is assumed that an autorun script will reboot the player once it performs necessary recovery tasks (i.e. the player will not automatically reboot).
- If the request handler responds with an HTTP code 204, or an HTTP code 200 containing an empty message body, the player will immediately reboot.
- If the request handler does not respond, the player will continue to send requests for 30 minutes before rebooting. Alternatively, a reboot will occur after only 5 minutes if the autorun script has encountered an `autorun-runtime-error`. This process will cycle indefinitely until an autorun script is received from the request handler.



Player Registry Entries

The following player registry entries are associated with the provisioning and recovery process. These keys can be modified by accessing the "networking" section using the `roRegistrySection` BrightScript object.

Registry Key	Description
a	The name of the <i>account</i> to which the player belongs

u	The name of the <i>user</i> in the account to which the player belongs
p	The account <i>password</i> that allows the player to access the request handler
g	The <i>group</i> within the account to which the player belongs
ub	The URL prefix that is applied to the URLs of the <i>ru</i> , <i>eu</i> , and <i>cu</i> key values. This prefix is not applied to a key value containing a colon.
ru	The provisioning/recovery URL used by the player to download an autorun script
eu	The URL to which error notifications are posted
cu	The URL to which crash reports are posted

Note: *The account, user, password, and group registry keys are used for provisioning/recovery with the BrightSign Network. A request handler may appropriate these keys, but they are not required.*

The following player registry entries are utilized by standard scripts. They may or may not be used as part of the provisioning and recovery process:

Registry Key	Description
nu	The URL from which to request the next sync spec (see the BrightSign Sync Spec technical note for more details)
tz	The time zone assigned to the device (see the <i>roSystemTime</i> entry in the BrightScript Object Reference Manual for available time zones)

URL Request Parameters

The following tables describe the parameters that are used for communicating between a player and the request handler.

Player Request Parameters

The following parameters are included in the header of HTTP GET requests that a player sends to the provisioning/recovery URL:

Parameter	Example Value	Description
Account	bsaccount	The <i>a</i> registry value described above
User	bsuser	The <i>u</i> registry value described above
Group	bsgroup	The <i>g</i> registry value described above
Password	bspassword	The <i>p</i> registry value described above
DeviceModel	XD1230	The model designation of the BrightSign player
DeviceFamily	cheetah	The family designation of the BrightSign player
DeviceId	X2F2CU0000065	The unique serial number of the player
DeviceFwVersion	4.7.146	The version of firmware currently installed on the player
DeviceUpTime*	240	The amount of time (in seconds) since the player booted up
RecoveryMode*	override	The mode of the current URL request: <ul style="list-style-type: none"> <i>override</i>: The initial HTTP GET request of a player with a valid autorun script <i>periodic</i>: Subsequent HTTP GET requests of a player with a valid autorun script <i>last-resort</i>: The HTTP GET request of a player without a valid autorun script
CrashDump*	yes	A flag indicating that the current boot was initiated by a

		forced reboot due to a crash (this header is only present if true)
StorageStatus	usb1=none;sd=autorun;sd2=storage;ssd=none;	The current autorun and file-system status of each storage device.

**These parameters are implemented in firmware versions 4.8.80 or later.*

Storage Status Parameters

The `StorageStatus` value contains a list of key-value pairs indicating the status of each storage device. The following storage device keys are returned:

BrightScript Key	Storage Status Key
USB1:	usb1
SD:	sd
SD2:	sd2
SSD:	ssd

The following are possible status values that can be returned for each storage device:

Storage Status Value	Description
none	The storage device was not detected.
storage	The storage device is present, but an autorun script was not detected.
autorun	The storage device is present and contains an <i>autorun.brs</i> or <i>autorun.zip</i> file.
autorun-load-error	The storage device is present and contains an <i>autorun.brs</i> or <i>autorun.zip</i> file that failed to load.
autorun-runtime-error	The storage device is present and contains an <i>autorun.brs</i> or <i>autorun.zip</i> file that encountered a runtime error.
error	The storage device is present but failed a file-system check. This value also implies that attempts to repair the file system also failed.

Server Response Parameters

The header of a request handler response may contain a single parameter. This parameter can accompany an HTTP 204 or 200 response (even if the response body is empty).

Parameter	Example Value	Description
Retry-After	7200	The amount of time (in seconds) that the device should wait to make a periodic request to the provisioning/recovery URL. If this parameter is not provided, the player defaults to a request interval of 2 hours.

Request Handler Workflow

The following workflow diagram suggests a simple request handler implementation.

This workflow assumes implementation of the following:

- A database that can associate forced-recovery flags with an individual device or group of devices
- A provisioning script

- A recovery script that first reformats the storage device
- A recovery script that does not reformat the storage device

This workflow does not implement the following features:

- Different provisioning scripts depending on the `DeviceId` or `Group` (or other HTTP GET parameter)
- Multiple provisioning or recovery autorun scripts for different storage devices.

