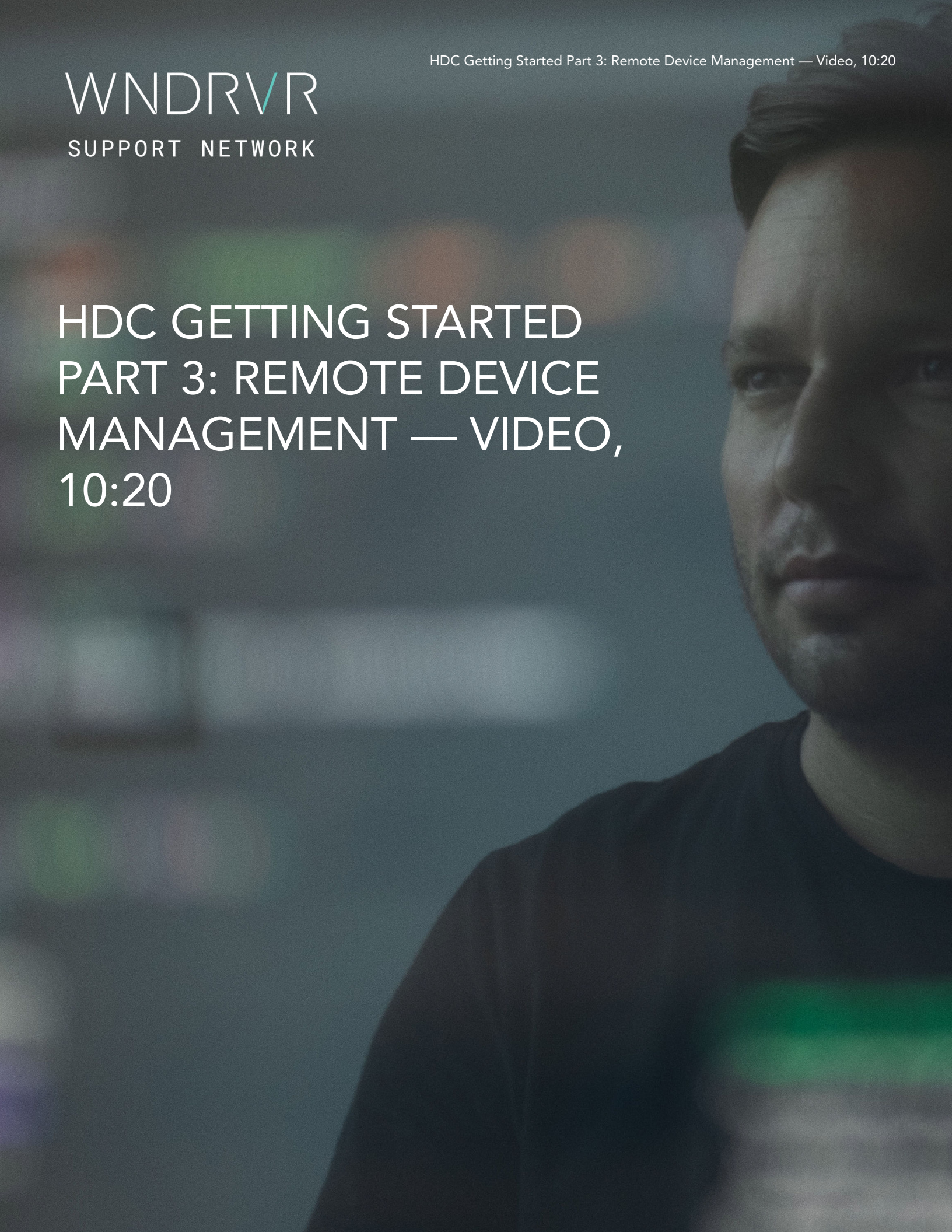
WNDRVR
SUPPORT NETWORK

HDC GETTING STARTED PART 3: REMOTE DEVICE MANAGEMENT — VIDEO, 10:20



Corporate Headquarters

Wind River
500 Wind River Way
Alameda, CA 94501-1153
U.S.A.
Toll free (U.S.A.): +1-800-545-WIND
Telephone: +1-510-748-4100
Facsimile: +1-510-749-2010

For additional contact information, see the Wind River website:

www.windriver.com

For information on how to contact Customer Support, see:

www.windriver.com/support

HDC Getting Started Part 3: Remote Device Management — Video, 10:20

29 January 2019

1. HDC GETTING STARTED PART 3: REMOTE DEVICE MANAGEMENT — VIDEO, 10:20

Published on 6 October 2015

Transcript

Time (mm:ss)	Narration
From: 00:12	Up until now we've been looking at some of the low-level capabilities in the device. When you first turn a device on, it will contact the cloud, it will boot up, and register with the cloud platform. But we looked at how to prepare the low-level build for that and also looked at some of the capabilities of writing scripts and writing applications to run on the device, but now let's take a step back and let's look at remote device management. This is the common use cases that come up when managing a device, and these apply regardless of application or application domain, vertical industry. It does not matter if its transportation, or medical, or factory automation. These use cases still apply, regardless. And so when a device first is powered up, it's called provisioning, and then
From: 01:02	sending down a configuration in a secure manner, performing maintenance which could include also software updates, sending out new packages, and also diagnostics. When something is going wrong, how to identify what the problem is. And then we're going to also take a look at decommissioning and billing issues that can come up as well. If we take a look at the first use case, this is a provisioning of a deployed device, and we kind of have seen this already in pieces at least, where when a device is first prepared, we set the initial configuration with the iot-config script, and then when that device is delivered to the end customer, and they first turn it on, that device will essentially make a phone home call and then contact the the Helix Device Cloud, and from there it can receive additional configuration.
From: 02:00	There may be specific application components, or additional configuration data that needs to be pushed down to it, but by simply identifying itself by its model type and serial number, the cloud platform can know what additional configuration data it might need, and you can kind of see the typical workflow here. On first boot, with just a factory default configuration, you know it can announce itself to the cloud, and then it will get its new configuration, and it may or may not need to restart the agent. It may or may not need to reboot itself. Those are optional things that may be defined by what that initial configuration that's being sent down requires. But then we've already seen how data items can then be sent up to the cloud, and this can all be automated with no human interaction so, this is really the ideal goal of the Internet of Things types of devices because it reduces the amount of field maintenance that's necessary.
From: 03:02	And we've also looked a little bit at some of these capabilities but, if you look at the ability to send files up and down to define alarms, to set data attributes, but we can also define schedules and we can specify whether it, you know, a rule about that device, in what condition or state its in. We can report status, and we can even report version numbers of installed packages as part of the data attributes that we send up. Also from the cloud side, we can specify actions to either restart the device or restart the agent, software running on the device, and we can execute some scripts and actions on the device, and we can initiate these things from the cloud side, and we can push down, you know, changes to certain data attributes such as the ping rate, or some kind of synchronization features that are necessary. The secure configuration, we've seen that when it when the device first
From: 04:04	boots up it establishes a secure tunnel back to the cloud, and so it is possible then that this communication is done in a secure way, and so you can see what that initial configuration setup looks like. We've already kind of looked at this a little bit, but if you look at the

	"default_settings" file, you can see what that initial configuration would look like. There are very few settings that need to be part of the, what's called the factory default settings, but the applications are installed in the /usr/bin directory. The SSL certificate is installed in the /etc/wra directory, and then the additional logging information and other files are provided. Any scripts or applications that are downloaded by default will go into the /wra/files/default directory. You can specify to move things to other locations, but once it, you know it connects to the cloud, then it can be pushed down it to the device.
From: 05:02	You can see here that when the device first boots up, it will start up the agent and the agent is in the /usr/bin directory called wra, and it's just passing an argument to which settings file to use. It can use the default_settings or it can point to some other settings file if necessary. We've also noticed that the API is dynamic. So we're defining these data items dynamically. There's no static configuration and we're not restricted in terms of the data items, so we can create new data items at will. The configuration also contains besides the initial server URL, the port, the ping rates. These are all the kinds of data that's in the conditional configuration file, and whether or not to include some remote session capabilities, and server certificates that are necessary. So, all of these things are defined in that one
From: 06:01	JSON configuration file that we've looked at. And then the solution flexibility is really what the agent and the cloud capabilities provide us. We find that many customers want to use the management capabilities of the cloud platform, but they want their data going to a separate direction. Maybe a proprietary off-loaded server or storage platform of their own, and so we can separate out the management and the data planes, simply by running a separate agent. We can run multiple agents on the device at the same time, and send the data to a different place than what we would do the management configuration. We can also route the data through the cloud platform, through our agent, and then redirect it out to an alternate storage location. And then one more configuration option, is that we can register applications with the agent
From: 07:04	that can then manage that private data cloud, but then still be able to report the logging information and event information up into the management cloud platform. So, and then in the maintenance scenario, we'll talk about here this is where you might want to define software updates. When a software package is delivered, maybe at some time later, it's necessary to upgrade that software. Maybe a problem has been reported by customers and it needs to be upgraded. So, this is part of the the lifecycle management of a device, and sending out software upgrades is certainly something that is supported in the platform. And then configuration management as well. We've looked at writing applications that go on the target, but there's also the ability to implement applications
From: 08:01	on remote servers that leverage the API of the cloud platform, and then there's also the ability to define custom objects and rules that are loaded into the platform itself. It's an extensible platform that provides the ability to define some extensions into the platform itself. And then remote diagnostics is a capability, where when something is going wrong or there is a problem with the device, you can go and examine the device. You can check some of the data attributes that have been reported, or you can determine if it's no longer communicating, or you can also establish a remote session into the device and get a shell, a console, into the device and actually run some diagnostics software remotely as well, so you can see that we'll simply open up a remote shell just like you were, you know an SSH or a telnet type of shell and you can connect into the device that way.
From: 09:01	This is all managed through a secure tunnel, so it's not open and you know hackable from the outside. Decommissioning is when you've determined that for some reason, the device needs to go, taken offline or out of service. This may be because of malfunctioning kind of scenario, or simply there are circumstances in the installation where it needs to be changed out, a new model needs to be replaced, or whatever, so you might determine to replace a device. And then there's also the the scenario where you can, you may want to support a billing capability, and this would be, perhaps, if you have a metered communication channel. Perhaps you're only communication link up to the cloud is through a wireless network that is metered, and you want

	to manage your billing for that device, or you want to provide also a billing capability to an end customer, and so
From: 10:04	there could be a framework installed to support billing for that scenario as well. In this session, we've kind of taken a high-level walkthrough of remote management capabilities. In the next section, we're going to take a look at implementing some of those capabilities in our device.

Contact: nlyons

Content ID: 045830

template('WindRiver/function/JS/wrConditionalContent');