
# netReady™

## Publisher User's Guide

## Version 2.0.7

## For Use With Publisher Version 2.0.14 or Greater

UpdateLogic Incorporated
508-624-8688 (TEL)
508-624-8686 (FAX)
http://www.updatelogic.com

Thank you for using the UpdateTV netReady™ software update solution. UpdateLogic Incorporated provides a comprehensive, cost-effective mechanism, which can update the complex firmware within a digital television by sending firmware updates over the Internet, digital broadcast infrastructure, and via USB.

If you find any problems with this software, please report them via email to techsupport@updatelogic.com.

Please visit our website at www.updatelogic.com for the latest news and information.

UpdateLogic Incorporated

508-624-8688 (TEL)

508-624-8686 (FAX)

contact@updatelogic.com

# Table of Contents

# 1 NetReady Publisher Utility

This document describes the operation of a command line application called Publisher that may be run under Windows or Linux. Publisher is a utility that has the following functions:

- Associates binary update components with compatibility parameters and *meta-data*, encrypts those components, and stores the encrypted components and their compatibility associations and meta-data in an archive CAB file for:
    - delivery to the UpdateTV Network, including internet, USB, broadcast, and UpdateTV Lab servers, via the UpdateTV NOC.
    - use with the UpdateTV TestCarouselBuilder utility to create test TS files when developing a broadcast-capable NetReady Agent for a given device.
    - use with the UpdateTV TSFactory tool to create TS files to be streamed to TV's in the factory for final update prior to shipping.
- Uses field-strength encryption on the binary update components for secure transmission over the UpdateTV broadcast network, internet, via USB, or optionally uses factory-strength encryption (faster decryption) for broadcast transmission in the factory.
- Manages the organization of the binary update components into transport modules which are easier to decrypt and more efficient to transmit on the broadcast network.

The compatibility information that Publisher associates with software update images is used by the NOC to provide the network "signaling" which allows NetReady Agent equipped devices to choose which software updates they're interested in.

## 1.1 Change Tracking

| **Version Number of This Document** | **Changes** |
|---|---|
| 2.0.7 | Command directive change and addition of production directive. |
| 2.0.6 | Removal of cdinterval command which is no longer needed. Switch to netReady from UpdateTV where appropriate. |
| 2.0.5 | Addition of new commands. |
| 2.0.4 | General cleanup. |
| 2.0.3 | -version enforcement. New test pattern for 2.0.3 Agent. –prevent_adds directive. –export directive. |
| 2.0.2 | Skipped. |
| 2.0.1 | Minor fixes. Addition of -test command. |
| 2.0.0 | Adapted from version 1.9.6 of this document for Publisher 2.0 |

## 1.2 Scope

This document assumes that the reader is familiar with the UpdateTV broadcast, internet, and USB delivery modes, the NetReady receiver Agent, and SDK tools. If you are not familiar with these items then please see the appropriate documentation as described in the *Related Documents*

section below. Readers should be well versed in digital television technology including digital receiver architecture, MPEG protocols, and embedded software development.

**Note:** Each release of this document is coordinated with a specific SDK release. Make sure you are working with the appropriate SDK release.

## 1.3 Audience

This document is designed to be read by engineers involved in managing the software update process of the devices they manufacture.

## 1.4 Related Documents

All NetReady documents are distributed with the NetReady SDK. The NetReady SDK is distributed with documentation that appeals to a wide audience. Only the most technical staff members need to read all the documentation.

| Document | Topic | Source |
|---|---|---|
| netProvision Technical Guide | Description of netProvision function and connected device integration. Useful to all audiences. | ULI |
| netReady Intro | General introduction to the UpdateTV technology. Useful to all audiences. | ULI |
| netReady Agent Integration Requirements | Target software and hardware requirements for your netReady device. | ULI |
| netReady Agent Porting Instructions | Step by step porting instructions for the Agent. Useful to receiver Agent porting engineers. | ULI |
| netReady Agent Validation Guide | Description of tests the CEM should run to validate their port of the netReady Agent to a CTV. Useful to all audiences. | ULI |
| netReady Factory Process Guide | Introduces netReady technology to your factory staff prior to production | ULI |
| netReady Technical Support Guide | Introduces netReady technology to your technical support staff prior to production | ULI |
| netReady NOC User's Guide | User's manual for interacting with the Network Operating Center via its web interface. Useful to carousel development and test engineers. | ULI |
| netReady Publisher | User's manual for the Publisher utility. Useful to all audiences. | ULI |
| netReady Technical Support Guide | An introduction to netReady and supporting CTVs for a CEM's technical support organization. | ULI |
| netReady Factory Process Guide | An introduction to netReady and manufacturing CTVs for a CEM's factory staff. | ULI |
| UpdateTV TestCarouselBuilder | User's manual for the TestCarouselBuilder utility. Useful to all audiences. | ULI |
| UpdateTV TSFactory | User's manual for the TSFactory utility. Useful to technology evaluators and engineers supporting factory updates. | ULI |
| UpdateTV StreamViewer | User's manual for the StreamViewer utility. Useful to all audiences. | ULI |
| UpdateTV Factory Update Guide | A document that explains the use of factory mode from an operational and programmatic point of view. Useful to technology evaluators and engineers supporting factory updates. | ULI |
| UpdateTV A/97 Implementation Specification | UpdateTV implementation of A/97 download data service. Useful to technology evaluators and advanced users. | ULI |
| ATSC A/97: Software Download Data Service | ATSC spec. Useful to technology evaluators and advanced users. | ATSC |

| | | |
|---|---|---|
| http://www.atsc.org/standards.html | | |
| ATSC A/90: Data Broadcast Standard<br>http://www.atsc.org/standards.html | ATSC spec. Useful to technology evaluators and advanced users. | ATSC |

## 1.5 Terms

| Term | Description |
|---|---|
| **API** | Application Programming Interface. |
| **ASI** | Asynchronous Serial Interface, often used to distribute MPEG transport streams. |
| **ATSC** | Advanced Television Standards Committee. www.atsc.org |
| **Back End** | MPEG decode, graphics and user interaction portion of the digital receiver hardware. |
| **BEM** | Broadcast Encoder Monitor. A 1U rack mount device that accepts ASI or SMPTE DTV MPEG data in and inserts the UpdateLogic broadcast carousel. Also contains an RF input to monitor the resulting output. This device is installed at both broadcast sites and in customer labs. |
| **CableLabs** | Research and development consortium that is dedicated to helping its cable operator members integrate new cable telecommunications technologies. www.cablelabs.com |
| **CAB File** | An archive file like a ZIP file that contains one or more files in compressed format. |
| **CEM** | Consumer Electronics Manufacturer |
| **CHM** | Customer's Hardware Model |
| **Component** | A separately updateable element within an update. |
| **CMG** | Customer's Model Group |
| **Content Service Provider** | Content Service Provider. For instance, Netflix, Vudu, Amazon, CinemaNow, etc. |
| **Connected Device** | A device such as a TV or Blueray player that supports an Ethernet and/or WiFi connection to the internet and uses that connection to connect to services like streaming media, news, weather, etc.. |
| **COUI** | Customer's Organizational Unique Identifier |
| **CSP** | Content Service Provider. |
| **DCR** | Digital Cable Ready. This acronym refers to consumer electronic devices equipped with Digital Tuners and Cable Card slots for use in receiving one-way Digital Cable signals and features. |
| **Front End** | Tuner and demodulation portion of the digital receiver hardware. |
| **Hardware Model** | A number describing a sub-set of a model group that is usually panel size specific. |
| **HM** | Hardware Model |

| | |
|---|---|
| **HMR** | Hardware Model Range |
| **Image** | An entire software or data entity that is broken into pieces called “modules” when broadcast on the network. |
| **Manufacturers’ ESN** | A serial number that is readable by software and also printed on a label affixed to the outside of a TV. Used by the consumer to identify their TV to technical support staff. Each CEM has a different serial number scheme. The customer can display the serial number by using the TV’s menus. Also called “the SN”. |
| **Manufacturer’s Software Version** | A string that contains version information like “TV_System_3.7”. The customer can display the software version using the TV’s menus. |
| **Model Group** | A number that uniquely defines a particular TV model within a give OUI (company). A model group may be sub-divided into different hardware models. |
| **MG** | Model Group |
| **Module** | Name of the distribution unit that a software or data “image” is broken down into in order to broadcast it efficiently on the network. |
| **MV** | Module Version |
| **netProvision** | A facility that is part of netReady used for delivering streaming media security credentials to a TV. |
| **netReady** | A suite of connected device services that includes netUpdate, netProvision, netRegister, and netDiag. |
| **netReady Agent** | Software embedded in a digital TV receiver to enable the reception of software updates via the Internet, Broadcast, and USB. |
| **netUpdate** | A facility that is part of netReady used for delivering security credentials to a device in the consumer’s home. |
| **NOC** | The UpdateTV Network Operations Center. A server farm that distributes updates over the Internet and broadcast networks. |
| **NOC Web Interface** | A web-based application that is used to manage updates and provisioning over the UpdateTV Network. The URL for this application is extdev.updatelogic.com for the development NOC and support.updatelogic.com for the production NOC. Often called simply “the NOC”. |
| **OAD** | Over Air Download (update via terrestrial RF broadcast). |
| **OOB Testing** | Out Of Box Testing |
| **Open Cable** | Standards organization setting standards for Open Cable networks and related technology.<br>http://www.opencable.com |
| **OTA** | Over The Air (i.e. Terrestrial Broadcast). |
| **OUI** | Organizational Unique Identifier |
| **PSI** | Program Specific Information. Generally refers to the PAT, PMT, CAT, NIT, and TDT tables in an MPEG |

| | |
|---|---|
| | stream. |
| **PSIP** | Program and System Information Protocol. Generally refers to the STT, MGT, VCT, RRT, EIT, ETT, and DCCT tables in an MPEG stream. |
| **Publisher** | A command line tool provide by UpdateLogic for the purpose of packaging a set of modules for software distribution. |
| **Registration** | The process that all TVs go through when they first connect to the Internet after shipment in order to register their identity with the UpdateLogic NOC and get a network identity. During this process the device's serial number is sent to the NOC and it receives a ULID that is used for all subsequent conversations with the NOC. |
| **Re-registration** | A process that can be initiated via the NOC web interface by support staff in order to force a TV to re-register and receive its security credentials again. |
| **Registration** | The process that all TVs go through when they first connect to the Internet after shipment in order to register their identity with the UpdateLogic NOC and get a network identity. During this process the device's serial number is sent to the NOC and it receives a ULID that is used for all subsequent conversations with the NOC. |
| **RTOS** | Real Time Operating System. |
| **Security Credential** | A TV-unique or model group-common file that a TV must be provisioned with in order to use the associated streaming service that relies on it. |
| **Streaming Service** | An application that provides streaming video or audio. |
| **StreamViewer** | A software tool supplied by UpdateLogic to view and analyze file based transport streams that contain compatible carousels. |
| **SV** | Software Version |
| **SVR** | Software Version Range |
| **Update** | An entity containing multiple components that may be software or data that replaces the components that were shipped with the TV. Used to fix bugs and add features. |
| **UpdateTV Agent** | Software embedded in a digital TV receiver to enable the reception of software updates via the UpdateTV Network. |
| **UpdateTV Network** | A collection of networked data insertion servers, and related broadcast equipment, used to distribute software updates to UpdateTV enabled receivers. The network provides for distribution via over-the-air and Open Cable digital networks environments. |
| **UpdateTV Receiver** | A digital television receiver enabled with the UpdateTV Agent so as to receive software updates from the UpdateTV Network. |
| **UpdateTV Server** | A computing device used to generate a data stream that can be multiplexed with other digital program content at a |

| | |
|---|---|
| | broadcast facility for inclusion in the ATSC broadcast stream. |
| **TestCarouselBuilder (TCB)** | A Win32 GUI based tool that creates test carousel transport streams from template test files and Publisher packaged software updates. |
| **Unit Under Test (UUT)** | The CEM's target receiver that is being tested. |

# 2 Using Publisher

## 2.1 General Considerations

Publisher takes a software image and associates it with compatibility information consisting of OUI, Model Group, Software Version Range, and Hardware Model Range and stores this compatibility information in a file called Publish.xml which is then archived with the software image in a CAB file. Additionally *meta-data* consisting of text and attributes is associated with the update components and stored along with the update binary. This

Every invocation of Publisher requires that you specify all of the compatibility information (OUI, MG, SW version range, and HW model range) as well as a -path argument specifying where to find the image to publish, a –cab argument specifying where to place the output CAB file, and a user name and password for accessing the UpdateTV NOC. There are a number of other optional arguments as well.

The Publisher command line interface was chosen to simplify the integration into a batch driven release process. You are encouraged to use script files to reduce typing errors and provide a record of the chosen compatibility parameters for a given Publisher invocation.

The compatibility parameters that you provide to Publisher along with the module names are transmitted on the UpdateTV network. This is called "signaling". This signaling is the information that the NetReady Agent running on your receiver uses to make a decision on whether to accept a module for download or not. Please see the *UpdateTV Update Organization Guide.pdf* document for a complete description of how modules are identified on the network and how you can develop compatibility strategies.

The OUI, Model Group, Software Version range(s), and Hardware Model range(s), must have all been created on the UpdateLogic NOC *prior* to the use of Publisher. Please see the document *UpdateTV Network Operations Center CEM Users Guide* for instructions on how to create and maintain these parameters on the NOC.

Publisher *must* be run on a Win32 PC that is connected to the internet. Publisher uses an internet connection to contact the UpdateTV NOC to retrieve encryption keys. The internet connection requires a valid UpdateTV user name and password. Please contact UpdateTV technical support to obtain your user name and password if you don't already have one.

Publisher accepts two types of software update images as inputs: a directory containing one or more modules that make up a software image that has been pre-split by the CEM or a file name that Publisher will automatically split. The –path argument specifies either a directory if the –split argument is *not* used or a file if the –split argument *is* used. The directory or file reference may be an absolute or relative path.

The output CAB file is given the name OUI-Model Group-YYYYMMDD-HHMMSS.cab. The output path that the –cab argument references *must* exist before invoking Publisher. Invoking Publisher with twithout arguments displays the following information:

```
Publisher.exe v2.0.14
Error:  Please supply a password. Use -pswd <password>.
Use --help to get a list of options.
Publisher failed!  Return code -1.
```

Invoking Publisher with the –-help argument displays usage information as shown here:

```
Usage: Publisher.exe [options] file1 file2 ....

  Tool Use Directives:
      -h, --help                    This help message
      -q, --quiet                   Squelch all output
      -script <script_file>         Accept further commands from the specified script file
      -break                        Stop processing script commands here, but continue
                                    executing.
      (note: all values are assumed decimal unless prefixed with 0x for hex and 0 for
             octal

  Global Update Directives:
      -o              <oui>         CEM's OUI
      -mg             <model group> Target platform's model group
      -version        <ULI version> Update's ULI version number.
      -in             <path>        Path to components referenced by -file directives
      -out            <path>        Output .CAB of .UTV file to <path>
      -user           <name>        Username used to connect to UpdateLogic
      -pswd           <password>    Password used to connect to UpdateLogic
      -hm <min max>                 Update's hardware model version ranges.  May be more
                                    than one.
      -for_hardware_models <N N>    (same as -hm).
      -sv <min max>                 Update's software version ranges.  May be more than
                                    one.
      -for_software_versions <N N> (same as -sv).
      -text_id <id>                 Defines a text ID string
      -text_def <id> <text>         Associate the specified id with the specified text at
                                    the update-level
      -lockf <file> <rtime file>    Create a lock entry for a given file.  Specify its
```

```
                                publish-time and run-time locations.
    -export                     Export a component manifest and UTV file
    -cab_name                   Specify the name of the cab file that is created
    -utv_name                   Specify the name of the exported UTV file
    -cm_name                    Specify the name of the exported component manifest
                                file
    -define <identifier> <def>  Specify a pre-processor definition
    -ppdbg                      Turn on pre-processor debugging
    -x                          Use external encryption
    -factory1                   Use factory strength level 1 encryption
    -factory2                   Use factory strength level 2 encryption
    -factory3                   Use factory strength level 3 encryption
    -field_test                 Sets flag that says this package is part of a field
                                test.
    -factory_test               Sets flag that says this package is part of a factory
                                test.
    -command                    Sets flag that says this package contains embedded
                                platform commands.
    -nosplit                    Doesn't split the components into modules.
    -splitsize     <n>          Module size in bytes (default 2mb)
    -repeats       <n>          Number of times to repeat modules in carousel
    -startdelta    <n>          Delay between transmission of one module and the next
    -cdrepeats     <n>          Number of repeats for component directories
    -prevent_adds               Sets flag that says next update cannot add components.

Component Directives:
    {                           Start a new component definition (nesting not allowed)
      Required -
        -name <component name>          Component's name string.
        -file <file_name>               Component's image file.
        -test <component size>          May be substituted for -file. Creates a test
                                        pattern component of the given size.
        -version <ULI version>          Component's ULI version number.

      Optional -
        -o <oui>                        Specify an OUI other than the platform OUI.
        -mg <model group>               Specify a model group other than the platform
                                        model group.
        -text_def <id> <text>           Associate the specified id with the specified
                                        text for this component
        -hm <min> <max>                 Specify hardware model range for this component.
        -for_hardware_models <N N>      (same as -hm).
        -sv <min max>                   Component's software version dependency on
```

```
                                    currently installed version of itself.  May be
                                    more than one.
      -depends_on <name min max>    (same as -sv except takes name of component that
                                     it is dependent on).
    Attribute Flags -
      -optional                     Sets flag that says this component is optional.
      -reveal                       Sets flag that says this component should be
                                    shown in a query.
      -store                        Sets flag that says this component should be
                                    stored not installed.
      -partition                    Sets flag that says this component is a
                                    partition rather than a file.
      -agent                        Sets flag that says this component contains the
                                    ULI Agent.
      -reboot                       Sets flag that says this component requires a
                                    reboot after install.
      -cem_attributes               Sets CEM flags for custom application.
      -no_validate                  Sets flag that says this component should not be
                                    hash validated if it's a partition.
      -index                        Sets flag that says this component is an
                                    indexable partition.
  }                           End current component definition

Publisher completed successfully.
```

## 2.2 Examples

Although Publisher will execute all commands directly from the command line the use of script files simplifies Publisher use. All of the following examples will show the contents of an example script file called *script.txt* .but the script file could be called anything. The syntax for using a script file is shown here:

```
publisher –script script.txt
```

```
The '#' is a comment in this script language.  Anything to the right of
the '#' is ignored.
```

### 2.2.1 A Single Component Example Script File

Although Publisher will execute all commands directly from the command line the use of script

```
# REQUIRED: basic NOC login credentials
–user myusername –pswd mypassword
# REQUIRED: Platform OUI and Model Group for the target device
```

```
-oui 0xFFFFFF -mg 0xFFFE
# REQUIRED: ULI version of this UpdateLogic, Inc.
-version 1
# REQUIRED: where Publisher will get files and where it
#            will output files
-in In -out Out

# REQUIRED (at least one): platform hardware model target
#                          compatibility range(s)
-for_hardware_models 1 3
# REQUIRED (at least one): platform software version target
#                          compatibility range(s)
-for_software_versions 1 1

# REQUIRED (at least one): component definition
# The open curly bracket indicates the start of a new component
# components can not be nested.
{
      # REQUIRED: the name of the component
      -name "cramfs"
      # REQUIRED: the file in the path specified by the "-in" directive
      #           that contains the component
      -file "cramfs.img"
      # REQUIRED: the version of this component
      -version 2
      # REQUIRED: a component-level software version target
      -for_software_versions 1 1

# The close curly bracket indicates the close of a component
# definition.
}
```

In this example the arguments have the following meaning:

- -user specifies that the user name used for the UpdateTV Web Service logon should be “myusername”.
- -pswd specifies that the password used for the UpdateTV Web Service logon should be “mypassword”.
- -oui specifies that the OUI is 0xFFFFFF

- -mg specifies that the model group is 0xFFFE
- -version specifies the ULI version of this update.
- -for_hardware_models at the update level (outside of a component definition) specifies that the hardware model of the targeted device may range from 1 to 3.
- -for_software_versions at the update-level (outside of a component definition) specifies that the platform software version of the targeted device should 1.
- -in indicates that the component files for this update should be taken from relative path ‘./In’
- -out indicates the directory where the output cab file should be written.
- The opening ‘{‘ indicates the start of a new component definition
- The –name inside of the {} indicates the name of the component is “cramfs”
- The –file inside of the {} indicates the actual file on the dev system where Publisher is being run that contains the component is called “cramfs.img”.
- The –version inside of the {} indicates the version of the cramfs component update is 2.
- The –for_software_versions inside of the {} indicates the component targeted by this update should have a software version of 1.
- The closing ‘}’ indicates the close of the component definition.

Substituting a real user name and password results in the following output showing Publisher succeeded:

```
Publisher.exe v2.0.14
Retrieving model data from support.updatelogic.com .... done.
Using data for: FFFFFF
Using model data for UpdateLogic 0xfffe
Setting Module Version to 0x1.
Retrieving key data from support.updatelogic.com .... done.
Outputting files to cab: Out/ffffff-fffe-20090408-170035.cab
Processing: cramfs (In/cramfs.img) .... done.
Processing: compdir-01of01 (Out/compdir-01of01) .... done.
Constructing publish data .... done.
Publisher completed successfully.
```

The file ffffff-fffe-20090408.cab will have been places in the sub-dir Out.

If an unknown user or password is provided then Publisher responds as shown below:

```
Publisher.exe v2.0.14
Retrieving model data from support.updatelogic.com .... failed!
Error: Login failure for unknownusername.
Publisher failed!  Return code -1.
```

If a input path or missing component file is provided then Publisher responds:

```
Publisher.exe v2.0.14
Retrieving model data from support.updatelogic.com .... done.
Using data for: FFFFFF
Using model data for UpdateLogic 0xfffe
Setting Module Version to 0x1.
Retrieving key data from support.updatelogic.com .... done.
Outputting files to cab: Out/ffffff-fffe-20090408-170854.cab
Processing: cramfs (BadPath/cramfs.img) .... failed!
Error: Could not open the file
Publisher failed!  Return code 6.
```

If the software version range(s) specified have not been created on the NOC then Publisher responds:

```
Publisher.exe v2.0.14
Retrieving model data from support.updatelogic.com .... done.
Using data for: FFFFFF
Using model data for UpdateLogic 0xfffe
Setting Module Version to 0x1.
Error: Software version range 100 100 not found for this platform model
group.
       You must create software version definitions for this range on
the NOC
Publisher failed!  Return code -1.
```

If the hardware model range(s) specified have not been created on the NOC then Publisher responds:

```
Publisher.exe v2.0.14
Retrieving model data from support.updatelogic.com .... done.
```

```
Using data for: FFFFFF
Using model data for UpdateLogic 0xfffe
Setting Module Version to 0x1.
Error: Hardware model range 100 100 not found for this platform model
group
       You must create hardware model definitions for this range on the
NOC
Publisher failed!  Return code -1.
```

### 2.2.2 Multiple Component Example Script File

```
# REQUIRED: basic NOC login credentials
-user myusername –pswd mypassword
# REQUIRED: Platform OUI and Model Group for the target device
-oui 0xFFFFFF -mg 0xFFFE
# REQUIRED: ULI version of this UpdateLogic, Inc.
-version 1
# REQUIRED: where Publisher will get files and where it
#            will output files
-in In -out Out

# REQUIRED (at least one): platform hardware model target
#                          compatibility range(s)
-for_hardware_models 1 3
# REQUIRED (at least one): platform software version target
#                          compatibility range(s)
-for_software_versions 1 1

# REQUIRED (at least one): component definition
# The open curly bracket indicates the start of a new component
# components can not be nested.
{
      # REQUIRED: the name of the component
      -name “cramfs”
      # REQUIRED: the file in the path specified by the “-in” directive
      #           that contains the component
      -file “cramfs.img”
```

```
      # REQUIRED: the version of this component
      -version 2
      # REQUIRED: a component-level software version target
      -for_software_versions 1 1

# The close curly bracket indicates the close of a component
# definition.
}
# OPTIONAL: additional component definitions
{
      # REQUIRED: the name of the component
      -name "some other component name"
      # REQUIRED: the file in the path specified by the "-in" directive
      #           that contains the component
      -file "somethingelse.bin"
      # REQUIRED: the version of this component
      -version 2
      # REQUIRED: a component-level software version target
      -for_software_versions 1 1
}
```

## 2.3 Argument Descriptions

Publisher supports the following update-level (outside of {} component definition) arguments:

| Argument | Required | Parameter | Description |
|---|---|---|---|
| -o –oui | Yes | 0-0xFFFFFF | CEM's OUI |
| -mg | Yes | 0-0xFFFF | CEM's Model Group |
| -version | Yes | 0-0xFFFF | ULI version number |
| -hm | Yes | 0-0xFF 0-0xFF | Targeted Hardware Model Range, may be used multiple times |
| -for_hardware_models | Yes | 0-0xFF 0-0xFF | Same as -hm |
| -sv | Yes | 0-0xFFFF0-0xFFFF | Targeted Software Version Range, may be used multiple times |
| -for_software_versions | Yes | 0-0xFFFF 0-0xFFFF | Same as -sv |
| -in | Yes | Path | The path to get the component files from. |
| -out | Yes | Path | The path to put the output files in. |

| | | | |
|---|---|---|---|
| -user | Yes | String | User name for NOC login |
| -pswd | Yes | String | Password for NOC login |
| -production | No | None | Configures the update for production distribution. When this directive is used Publisher contacts the production NOC and the Agent query host directive is set to also contact the production NOC. Without this directive Publisher contacts the extdev NOC and instructs the Agent to contact the extdev NOC as well. |
| -text_id | No | String | Declaration of the ID string associated with a –text_def directive used at the update or component-level. |
| -text_def | No | ID String | Definition of a text string associated with a previously declared test ID. This text will be available to the Agent at the update-level. |
| -lockf | No | Publish-time file path, run-time file path | Specifies a file to check the hash of before allowing the active component manifest to be opened. |
| -export | No | None | Exports a UTV file and component manifest in addition to the CAB file. |
| -cab_name | No | Name of CAB file to be output | Normally Publisher creates a CAB file name from the OUI/MG and date/time. This directive allows the CAB file name to be specified. |
| -utv_name | No | Name of UTV file to be output | Must be used with the –export directive. Override the standard (OUI-MG.utv) file name. |
| -cm_name | No | Name of component manifest file to be output | Must be used with the –export directive. Override the standard (component.manifest) file name. |
| -define | No | Identifier, replacement | Replaces all instances of the identifier with the specified replacement text in all *following* occurrences of the identifier. |
| -ppdbg | No | None | Lists all replacements created by the –define directives. Useful for debugging define replacements. |
| -x | No | None | Enables hardware-specific encryption. This requires a special build of Publisher. |
| -factory1 | No | None | Specifies that the module(s) should be encrypted using factory-strength level *one* encryption. When –factory1 is used the model group specified with the –mg |

| | | | |
|---|---|---|---|
| | | | argument must be enabled for factory encryption on the NOC. Please see the *UpdateTV Factory Update Guide* for more information about factory mode. |
| -factory2 | No | None | Specifies that the module(s) should be encrypted using factory-strength level *two* encryption. When –factory2 is used the model group specified with the –mg argument must be enabled for factory encryption on the NOC. Please see the *UpdateTV Factory Update Guide* for more information about factory mode. |
| -factory3 | No | None | Specifies that the module(s) should be encrypted using factory-strength level *three* encryption. When –factory2 is used the model group specified with the –mg argument must be enabled for factory encryption on the NOC. Please see the *UpdateTV Factory Update Guide* for more information about factory mode. |
| -field_test | No | None | Sets a flag in the component directory of the update that indicates that the update is compatible with devices that likewise have the field_test flag set in their component manifest. |
| -factory_test | No | None | Sets a flag in the component directory of the update that indicates that the update is compatible with devices that likewise have the factory_test flag set in their component manifest. |
| -command | No | None | Tell the Agent that the update contains only commands, not an update. See the command section below. |
| -nosplit | No | None | Doesn't split the components into modules. By default each component is is split into 2MByte modules. |
| -splitsize | No | Size in Bytes | Specifies the maximum size in bytes of each of the modules created by the split process. Minimum of 2mb, maximum of 8mb.<br>(The size of the last module may differ). |
| -repeats | No | Number of Repeats | Specifies the number of repeats per module in a broadcast carousel. |
| -startdelta | No | Seconds | Specifies the number of seconds between the transmission of one module and the next in a broadcast carousel. |

| | | | |
|---|---|---|---|
| -cdrepeats | No | Number of repeats | Specifies the number of times a component directory is repeated in a broadcast carousel |
| -prevent_adds | No | None | Set's a flag saying that any future updates will not be allowed to add new components. All components must agree with the definitions provided in this update. |
| -align4 | No | None | Aligns UTV data structures on 4 byte boundaries. Must be used in conjunction with special Agent defines and is only needed for some processors to avoid long word access on non 4-byte address boundaries. |
| -h –help | No | None | Displays the usage information. |
| -q –quiet | No | None | Disables the display of any information during operation. |

Publisher supports the following component-level (inside of {} component definition) arguments:

| **Argument** | **Required** | **Parameter** | **Description** |
|---|---|---|---|
| -name | Yes | String | Name of the component. |
| -file | Yes | File Name | Name of the file that contains the component on the system where Publisher is run. |
| -test | No | File Length | Creates a test pattern file of the specified length in place of a real component for test purposes. Used instead of –file. Maximum test image size of 256mb. |
| -version | Yes | 0-0xFFFF | Version of this component |
| -for_software_versions | Yes | 0-0xFFFF 0-0xFFFF | Targeted software version range for this component. |
| -for_hardware_models | No | 0-0xFF 0-0xFF | Targeted hardware model range for this component. |
| -o –oui | No | 0-0xFFFFFF | Component OUI if different from platform OUI specified at update-level. |
| -mg | Yes | 0-0xFFFF | Component Model Group if different from platform Model Group specified at update-level. |
| -text_def | No | ID String | Definition of a text string associated with a previously declared test ID. This text will be available to the Agent at the component-level. |
| -depends_on | No | ComponentName | Depends on the named installed |

| | | | |
|---|---|---|---|
| | | 0-0xFFFF 0-0xFFFF | component being at the software version described. |
| -optional | No | None | Set's a flag saying this component is optional. By default the component is required. |
| -reveal | No | None | Set's a flag saying this component update should be presented to the user during any user update acceptance interactions. By default the component is not shown. |
| -store | No | None | Set's a flag saying this component should be stored and not installed directly. By default components are installed directly. |
| -partition | No | None | Set's a flag saying this component is a partition. By default a component is considered a file. |
| -agent | No | None | Set's a flag saying this component contains the UTV Agent. |
| -reboot | No | None | Set's a flag saying that after installtion of this component the system should be rebooted. |
| -cem_attributes | No | None | 16 bit value of flags that is made available to the Agent to mean anything the CEM would like. |
| -no_validate | No | None | Specifies that this component should not be validated with a SHA256 hash. |
| -index | No | None | Sets a flag that says this component is an indexable partition meaning it can be swapped. |

## 2.4 Text Directives

Publisher supports text directives that make text available to the Agent at both the update and component level. The –text_id directive declares a text ID string that will be referenced later by a -text_def directive. The –text_id directive is only an instruction to Publisher to allow references to this ID in later –text_def directives. It has not effect on the update itself. –text_def directives can occur at both the update-level (outside of {} component definitions) or at the component level (inside of {} component definitions). The text contained in the definition is made available to the Agent via the commands:

```
UTV_RESULT UtvPublicImageGetUpdateTextId( UTV_UINT32 hImage,
UTV_UINT32 uiIndex, UTV_INT8 **ppszTextIdentifier );
```

```
UTV_RESULT UtvPublicImageGetCompTextId( UTV_UINT32 hImage,
UtvCompDirCompDesc *pCompDesc, UTV_UINT32 uiIndex, UTV_INT8
**ppszTextIdentifier );

UTV_RESULT UtvPublicImageGetUpdateTextDef( UTV_UINT32 hImage,
UTV_INT8 *pszTextID, UTV_INT8 **ppszTextDef );

UTV_RESULT UtvPublicImageGetCompTextDef( UTV_UINT32 hImage,
UtvCompDirCompDesc *pCompDesc, UTV_INT8 *pszTextID, UTV_INT8
**ppszTextDef );
```

Please see the *UpdateTV Agent Design Specification* document for more information on these commands.

## 2.5 Update-Level Attributes

Publisher supports the update-level attribute flags described above. These attributes are set via the directives:

| |
|---|
| -ftest |
| -prevent_adds |

These directives cause the `pCompDirHdr pointer returned by the UtvImageGetCompDirHdr(UTV_PUBLIC_UPDATE_SUMMARY->`hImage ) call to be set. Please see the *UpdateTV Agent Design Specification* document for more information on these commands.

## 2.6 Component-Level Attributes

Publisher supports component attribute flags described above. These attributes are set via the directives:

| |
|---|
| -optional |
| -reveal |
| -store |
| -partition |
| -agent |
| -reboot |
| -cem_attributes |

The directives cause the `pCompDesc-> uiComponentAttributes` field of the `UTV_PUBLIC_COMPONENT_SUMMARY` to be set. Please see the *NetReady Agent Design Specification* document for more information on these commands.

## 2.7 Command Directives

Publisher supports 4 types of command directives. Command directives instruct the Agent to execute the specified command. Commands use the test directive syntax, but the text ID is a special identifier that causes the Agent to execute the text that it refers to. The syntax of a command directive is as follows:

```
-text_id  "CommandIdentifier_Index”
-text_def "CommandIdentifier_Index” "command”
```

Each CEM and model group is given its own unique command identifiers. You will receive your custom command identifiers directly from ULI. The identifiers fall into 4 types:

| Command | Description |
|---|---|
| USB Command | This type of command is executed once when it is detected during a USB insertion of an utv file that contains a “-command” directive. This utv file will *only* execute the commands it contains. It will not update anything. |
| Pre-boot Command | This type of command is executed once before an update is booted for the first time. The –command directive is *not* used with this type of command because it’s part of an update. |
| Post-boot Command | This type of command is executed once *after* an update is booted for the first time. The –command directive is *not* used with this type of command because it’s part of an update. |
| Post-every-boot Command | This type of command is executed *every time* the update boots. It’s useful for test looping. The –command directive is *not* used with this type of command because it’s part of an update. |

The actual commands that are executed are entirely dependent on the commands the target platform supports via its UtvPlatformCommandInterface() function. Typically this interface is given access to special debug commands and the Bash shell or equivalent. Commands often include the ability to delete and copy files, set special modes that may enable serial output or logging.

Note that the command index must start at zero and increment by one for each command. Here’s a set of sample command directives. In this case the example CEM would have received the USB command string “0119d7d92f3d” from ULI.

```
# tell the Agent that this update only contains commands
-command

# FIRST command: remove the active component manifest
-text_id  "0119d7d92f3d_0”
-text_def "0119d7d92f3d_0” “system rm
/mnt/persistent/UTV/component.manifest”
```

```
# SECOND command: copy a component manifest from the USB to the
active folder
-text_id "0119d7d92f3d_1"
-text_def "0119d7d92f3d_1" "system cp
/mnt/sda1/component.manifest
/mnt/persistent/UTV/component.manifest"

# THIRD command: display the active area
-text_id "0119d7d92f3d_2"
-text_def "0119d7d92f3d_2" "system ls /mnt/persistent/UTV"

# FOURTH command: issue a special debug command
-text_id "0119d7d92f3d_3"
-text_def "0119d7d92f3d_3" "dbgtrace on"
```

The above is an illustration of how to create a "command stick" where the sole purpose of the "update" is to contain commands to execute. There is no update. This is just one type of command that the Agent supports. The other three types of commands are executed as a part of an update or every time the device boots. They may be placed anywhere in an update script file *outside* of a component directive. You can place as many command directives in the update as you would like by using the indices.