Teleprint

Client Configuration Reference

Table of Contents

Introduction....3
Scope....3
Target Audience....3
Example Configuration....4
Module Types and Configuration Parameters....5
Raw Printer Module....5
Forwarder Module....5
Enabling the Forwarder Module....5
Serial Device Module....6
Test Device Module....6
Supervision Module....7
Supervision Commands....7
Spooler Module....8
Connect API Module....8
Websocket API Module....9
TLS Module....10
Connection Module....10
Launcher Sequence....11

Introduction

Scope

This document describes the configuration syntax for the Device Web Connector client component.

Target Audience

This document targets developers or administrators who wish to manually configure operation of the Device Web Connector client component. The target audience is expected to be familiar with basic network protocols and how they are layered on each other.

Example Configuration

Client connections are stored as JSON structures whether they are manually configured, imported, or launched from the custom URL scheme.

Below is an example of a printer configuration:

```
{
	"guid" : "{D47C3A99-DF0F-4A82-8756-3A271F56C416}",
	"modules" : [
		{
			"type" : "rawPrinter"
			"printerName" : "Zebra Direct"
		}, {
			"type" : "supervision"
		}, {
			"type" : "connectApi",
			"password" : "admin",
			"path" : "/bob_printer",
			"username" : "admin"
		}, {
			"type" : "tls",
			"host" : "print.company.com",
			"keepAlive" : 20,
			"verifyHost" : true,
			"verifyPeer" : true
		}, {
			"type" : "connection",
			"host" : "print.company.com",
			"port" : 443
		}
	]
}
```

The optional "guid" element is used to detect duplicates, and will be assigned if one is not provided. If you are sending configurations to remote users, it's probably good to assign this yourself.

The modules array is a list of modules which form a protocol stack, listed from the "bottom" up. The bottom module must always be a device module of some sort, and the top must always be a connection module.

Manual configurations can arrange intermediate modules in any order, although many orders wouldn't make sense. Configurations launched via the custom URL scheme are validated against a whitelist of valid patterns.

Module Types and Configuration Parameters

Modules operate independently of each other, so some items such as hostname are repeated. This is normal, because the same information is needed by multiple modules.

The following are the supported module types, along with the configuration parameters in their JSON configuration block:

Raw Printer Module

The raw printer module is a device module and therefore sits at the bottom of the stack. Instead of connecting directly to USB ports, this module takes incoming data and builds a print job to pass off to the operating system print spooler. The job is built with a RAW type so that the native printer language can be used.

Variable	**Type**	**Meaning**
type	String	Must be "rawPrinter"
printerName	String	Name of the printer

Forwarder Module

The forwarder sends data over a secondary TCP endpoint to a device on the local network. For security reasons this module cannot be instantiated as part of a web-managed connection.

Variable	**Type**	**Meaning**
type	String	Must be "forwarder"
host	String	Host or IP address to connect to
port	Number	Port number to connect to

Enabling the Forwarder Module

Use of the forwarder module requires the following key to be set to a non-zero DWORD:

```
HKEY_CURRENT_USER\Software\Teaglu\dtclient\allowForwarder
```

Serial Device Module

The serial module connects the upstream modules to a serial or "COM" port. Baud rate and other port settings can be included as part of the configuration or configured via an out-of-band message via the supervision module "connect" message.

Variable	Type	Meaning
type	String	Must be "serial"
port	String	Name of the COM port
speed	Integer	The port speed in bits per second.
byteSize	Integer	The number of bits per byte, between 6 and 8.
parity	String	One of the following parity values: none, even, odd, mark, space.
stopBits	Integer	The number of stop bits, which must be 1 or 2.
flowControl	String	One of the following strings: hardware, software, none.

Test Device Module

The test device pops up a window that lets the user send and receive data directly, for use as a debugging tool.

Variable	Type	Meaning
type	String	Must be "testDevice"

Supervision Module

The supervision module wraps the data stream in an IAC-like protocol to allow for passing out-of-band messages such as start and stop of job. The supervision module has no parameters other than the type.

Variable	Type	Meaning
type	String	Must be "supervision"
keepAlive	Numeric	Set a number of seconds for keepalive messages.

Each supervision message consists of a start-of-message character 0xFF. The second byte after the start character may be 0xFF to escape the original 0xFF character and send a literal 0xFF.

If the second byte is not the escape sequence, the lower four bits of the second byte indicate the number of additional bytes in the message, and the upper four bits of the second byte indicate the command.

Supervision Commands

Command	Value	Meaning
CONNECT	0x10	The other end of the connection has been connected to a new endpoint. This indicates start-of-job for printer data.
DISCONNECT	0x20	The other end of the connection has been disconnected from the current endpoint. This indicates end-of-job for printer data.
KEEPALIVE_REQUEST	0x30	The peer is requesting connection acknowledgment.
KEEPALIVE_RESPONSE	0x40	The peer is responding to a keepalive request.

Spooler Module

The spooler module is currently in testing.

The spooler module implements a buffer in either the upstream direction, the downstream direction, or both. Buffers are backed by disk file and can be up to 16GB in size per direction.

For each direction where buffering is enabled, writes will be added to the buffer file and picked up by a dedicated write-behind thread.

Variable	**Type**	**Meaning**
type	String	Must be "spooler"
upstreamSize	Numeric	Size of upstream buffer in bytes. If this value is zero or is omitted, no buffering is done in the upstream direction.
downstreamSize	Numeric	Size of downstream buffer in bytes. If this value is zero or is omitted, no buffering is done in the downstream direction.

Connect API Module

The connect API module implements using the HTTP CONNECT verb to ask a webserver for a direct connection to a resource.

Variable	**Type**	**Meaning**
type	String	Must be "connectApi"
path	String	The path to be used as part of the CONNECT verb, which typically indicates a path or junction point.
username	String	The username to use as part of basic authentication.
password	String	The password to use as part of basic authentication. If the username is not specified, this value is passed as is as a bearer authentication header.

Websocket API Module

The websocket API module implements the Websocket (RFC 6455) protocol to ask a webserver for a streamed connection.

Variable	**Type**	**Meaning**
type	String	Must be "websocketApi"
path	String	The path to be used as part of the GET verb during the initial protocol, which typically indicates a path or junction point.
username	String	The username to use as part of basic authentication.
password	String	The password to use as part of basic authentication. If the username is not specified, this value is passed as is as a bearer authentication header.
host	String	A hostname to pass as header during authentiation, which may be needed to determine the host in a virtual hosting setup.
origin	String	The origin to be used as part of the websocket protocol. **This is a required part of the protocol.**

TLS Module

The TLS module implements wrapping the data stream in the TLS protocol. The implementation uses the Windows SSPI interface to support custom root certificates used for SSL inspection.

Variable	**Type**	**Meaning**
type	String	Must be "tls"
host	String	Host name to pass as part of the SNI extension, which may be required to correctly access a server in a virtual hosting environment.
verifyPeer	Boolean	Requires that the remote host certificate is signed by a well-known authority. If this is set to false then self-signed and other invalid certificates are accepted. **This should not be set to false in production.**
verifyHost	Boolean	Requires that the remote host name match the name which was requested. **This should not be set to false in production.**

Connection Module

The connection module sits at the top of the protocol stack and implements a TCP connection to a remote host.

The connection will be continously re-attempted if an error occurs on a permanent connection. A connection error on an ephemoral connection will result in the connection being removed – this is a normal method of closing a web-managed connection.

Variable	**Type**	**Meaning**
type	String	Must be "connection"
host	String	Host name or IP to connect to
port	String	TCP port number to connect to.

Launcher Sequence

The launcher executable dtlaunch.exe is registered for two custom URL schemas: devtow: and devtows:. These correspond to un-encrypted HTTP and encrypted HTTPS. When the launcher is invoked, the following steps take place:

1. The dtlaunch.exe sends a POST request to the corresponding http/https endpoint which identifies the client version. The POST data will have the following format:

   ```
   {
         "version": "{client version}"
   }
   ```

2. If the response to the POST call does not have a status of 200, the program exits.
3. If the response is not a valid JSON object, the program exits.
4. If the response does not have a boolean attribute "valid" with the value true, the program exits.
5. The launcher displays a list of devices available for sharing.
6. If the user does not press the Allow button, the program exits.
7. The devices which the user authorized to share are sent as a POST request to the same endpoint as step 1, with the following format:

   ```
   [
         {
               "type": "rawPrinter",
               "printerName": "{name as it appears to the user}",
               "printerDriver": "{printer driver name as hint to backend}"
         },
         {
               "type": "rawSerial",
               "serialPort": "{COM port name}"
         }
   ]
   ```

8. The launcher interprets the response as an array of client configurations.
9. For each configuration, the launcher validated the configurations as follows:
   1. The arrangment of modules must match a built-in whitelist.
   2. Any modules which reference hostnames must refer to the same hostname as the original URL request.
   3. TLS checks cannot be disabled.
   4. Only devices which were authorized by the user can be referenced.
10. The dtclient.exe program is started if it is not already started, or if it is already started a handle to the running instance is found.
11. The configuration JSON is passed to dtclient.exe by means of a Windows message.