CHAPTER 1

Getting Started: Learning INI File Basics

In this chapter you will learn how to construct and use the supported INI files.

It includes the following sections:

After you become familiar with the INI file basics, you can refer to the parameter details you need in the other chapters and appendixes of this guide.

INI Parameter Updates Using Cisco VXC Manager

Cisco VXC Manager is the standard tool used to push and schedule INI configuration updates to your thin clients. Cisco VXC Manager allows you to push the common INI file configurations to all of the thin clients in your environment.

Caution Do not modify INI files directly on the Cisco VXC 6215 as doing so can cause configuration issues and operational errors.

For detailed information about configuring thin clients using Cisco VXC Manager, see *Administration Guide for Cisco Virtualization Experience Client Manager* and the *Cisco Virtualization Experience Client 6215 Administration Guide*.

Supported INI Files You Can Construct

The INI files contain the parameters (and associated options and values) necessary for the functionality you want.

You can construct the following INI files:

- wlx.ini file (see Working with wlx.ini Files, page 1-2)
- $MAC.ini file (see Working with $MAC.ini Files, page 1-2)

Working with wlx.ini Files

A wlx.ini file contains the global parameters you want that affect all thin clients accessing the server.

Working with $MAC.ini Files

A $MAC.ini file contains device-specific configuration parameters. If the thin client locates a $MAC.ini file (stored in the same directory as a wlx.ini file), the thin client uses the $MAC.ini file for its configuration rather than the wlx.ini file. In this case, the thin client does not access the wlx.ini file unless you specify the include=wlx.ini parameter in the $MAC.ini file.

Note If a duplicate parameter appears in both the wlx.ini and the $MAC.ini files, the thin client assigns higher precedence based on which parameter appears last in the $MAC.ini file: either the duplicate parameter or the include=wlx.ini parameter.

For example, assume the following parameter definitions:

- The wlx.ini file contains parameterA=valueB
- The $MAC.ini file contains parameterA=valueC

The following table describes the final value for parameterA based on its placement relative to the include=wlx.ini parameter in the $MAC.ini file.

Parameter placement	Final value for parameterA
parameterA=valueC appears **before** include=wlx.ini	ParameterA=ValueB (from wlx.ini file)
parameterA=valueC appears **after** include=wlx.ini	parameterA=valueC (from $MAC.ini file)

Rules and Recommendations for Constructing the INI Files

In general, Enhanced SLE INI files follow currently accepted standard INI file formatting conventions. The INI files consist of parameters that you can enter as necessary for reference. The parameters are not mandatory unless you require changes from the default values or the parameter is noted as required in the tables. Every parameter has a name and a value with the name appearing to the left of the equal sign (name=value). All parameters with the same name in the various INI files have the same meaning (that is, a parameter named XYZ in a wlx.ini file and named XYZ in a $MAC.ini file have the same meaning). Number signs (#) indicate the start of a comment. Comments can begin anywhere on a line. Everything between the # and the End of Line is ignored.

Along with these general formatting conventions, use the following guidelines when you construct Enhanced SLE INI files:

1. **Order of Parameters**

   List global connect parameters before other connect parameters in a wlx.ini file.

2. **Mandatory Parameters**

   As previously stated, you can enter parameters as necessary for reference, but the parameters are not mandatory unless you require changes from the default values or the parameter is noted as required in the tables.

   To create a connection using an INI file, you must specify the Connect= parameter, as well as the mandatory connect options associated with the specified connection type (for example, for Mozilla Firefox or VMWare View). For more information about the Connect= parameter, including configuration examples, see Connect Options, page A-1.

3. **Use of Backslashes and White Spaces**

   To indicate line continuation, place a space and backslash (\) at the end of a line. The backslash means that the line and the following line are, for the purposes of reading code, the same line. No white space can appear after the backslash; the requirement of white space between parameter entries is maintained by the use of the space before the backslash. In addition, starting all parameters at the left margin and placing at least one leading space (or tab) at the beginning of all (and only) continuation lines makes an INI file easier to read.

   Note that in circumstances where you require string concatenation, you can use a backslash without a space before or after it to concatenate with the first set of characters from the previous line; for example the strings snow and ball may be concatenated to give snowball.

   You must use the backslash continuation character to configure a connection with multiple parameters. For examples, see Connect Options, page A-1.

4. **Use of Blank Lines**

   Use blank lines to make the code easier to read.

5. **Use of Number Signs**

   As stated previously, number signs (#) indicate the start of a comment. Comments can begin anywhere on a line. Everything between the # and the End of Line is ignored.

6. **Use of Quotation Marks**

   You must place string parameters containing white spaces inside quotation marks (use common-practice nesting rules).

7. **Use of List Separators**

   Use semicolons or commas for list separators.

8. **Use of Equivalent Parameter Values**

   For parameter values of type {0, 1}, the 0 indicates false or no, and the 1 indicates true or yes, as applicable. You can use the format {0, 1} as an alternative to the format {no, yes}.

9. **Number of Connection Entries Allowed**

   The combined number of connection entries that you define in a wlx.ini file cannot exceed a defined total maximum number of connections. The maximum number of connections has a default limit of 216, but you can modify this limit within a range of 100 to 1000 using the wlx.ini file.