lxPatIO

lxPatio

Linux Pattern I/O stress test

Intel Corporation

lxPatIO

Table of Contents

Part I Introduction

1 Installation....7

Part II Subtests

1 Auto/Reconfiguration.... 9
2 Synchronous I/O.... 9
3 Asynchronous I/O.... 9
4 Completed I/O.... 9
5 Mapped I/O CopyFile....10

Part III Parameters

1 Client password.... 11
2 Client share.... 11
3 Client username.... 11
4 Collect stats.... 11
5 Delete old data files.... 11
6 Destination file size.... 12
7 Device nodes.... 12
8 Disable net.... 12
9 Exclude interface.... 12
10 Halt on error.... 12
11 Ignore boot drive.... 13
12 Interface range.... 13
13 Invert.... 13
14 Iterations.... 14
15 Maximum client count.... 14
16 Mount root.... 14
17 Offset.... 14
18 Outstanding IO ops.... 15
19 Pattern directory.... 15
20 Patterns.... 15
21 R-W completions.... 15
22 Random pattern seed.... 15
23 Randomize.... 16

24 Read-only..... 16
25 Retries..... 16
26 Runtime..... 16
27 Save file count..... 16
28 Statistics reporting..... 16
29 Sticky filename count..... 16
30 Target directory..... 16
31 Targets..... 16
32 Thread values..... 16
33 Transfer size..... 16
34 W-R ratio..... 17
35 Additional Client shares 17

1 Introduction

Version 2.13.1.0005

lxPatIO, short for "Pattern Input/Output", is a test module designed to write a pattern to specific targets, read the the pattern back from those targets, then compare the bytes read back with the original bytes written. The targets may be a combination of local hard drives, peripheral drives (e.g. USB thumb drives), optical drives (e.g. CD/DVD) and network shares.

How lxPatIO performs these write-read-comparisons is defined by the Subtests that are supported by this module. These subtests are useful for targeting your testing towards specific types of file operations that the operating system supports. Please read the Subtests section for more information on the supported subtests of lxPatIO.

A large number of parameters are provided to help target your testing to your specific needs. Please read the Parameters section for more information on the supported parameters of lxPatIO.

To report an issue with lxPatIO, please send an email to server.validation@intel.com

1.1 Installation

lxPatIO requires a test executor to be installed. This test executor is called iMTA Validation Stress Suite. You may have access to either just the test executor or perhaps you have access to the package containing the test executor and a handful of test modules (including lxPatIO). That package has a name in the form of IxVSS, where "x" is either "W" or "L", representing Windows or Linux, respectively. Those packages are designed to be installed through an installation process different from the one explained here, but once you install an IxVSS package, you automatically have lxPatIO installed as well.

Therefore, it is assumed you already have a test executor installed. If so, then simply place the binary in the same folder where the test executor resides.

Now that the lxPatIO module is installed, it is ready to be used by test packages. However, you will need to determine if package and/or snippet files contain a reference to the binary name for execution. In other words, you must open (with a text editor or with the test executor) the snippet(s) or package(s) that will make use of lxPatIO and confirm that the "Binary" field lists the full path to the binary.

1.2 FAQ

Answers to some commonly asked questions...

1. **How do I set up the network I/O tests?**
   Samba (CIFS) shares are required on clients connected to the SUT and will be used as read/write targets for the test. The clients can be Windows or Linux, but we've observed better throughput using Linux clients.

2. **How do I exclude network interfaces from being discovered during auto/reconfigure?** See the parameter Exclude interface for options on excluding network interfaces from discovery. Keep in mind that there is currently no support for netmasks. You can only enter specific IP addresses or an interface name. You should enter the information in the "Template Configuration" then save the package file.

2 Subtests

lxPatIO supports a list of subtests used to target your testing towards different I/O mechanisms supported by the operating system you are running your test on. However, the list below is a super-set of the tests supported as not all of the tests have been implemented.

lxPatIO is targeted to support the following tests...

Auto-configuration/Reconfiguration
Synchronous I/O
Asynchronous I/O
Completed I/O
Mapped I/O
CopyFile

2.1 Auto/Reconfiguration

Auto-configuration/Reconfiguration is a required subtest of any module CTC supports. It is designed to initialize test parameters based off of the platform configuration. If auto/reconfiguration fails, it is highly unlikely any other subtest will function correctly.

2.2 Synchronous I/O

The Synchronous I/O subtest is designed to perform reads and writes synchronously using the operating system's basic blocking "read()/write()" functions. The test will wait for a return value from these functions before progressing.

2.3 Asynchronous I/O

Also known as the "Overlapped I/O" test, the Asynchronous I/O subtest will perform the read()/write() operations but will not wait for the operation to complete. Instead, an event is created which will signal the completion of the operation. Therefore, you can expect this test to queue many operations before the first operation completes. The parameter "Outstanding IO ops" defines the limit of queued operations. These operations are not guaranteed to finish sequentially.

At the time of this writing, the Linux version of lxPatIO does not yet support the Asynchronous I/O subtest.

2.4 Completed I/O

The Completed I/O subtest is designed to behave very similarly to the Asynchronous I/O subtest. The only difference is the signaling mechanism used by the operating system to indicate the operations are completed.

At the time of this writing, lxPatIO does not yet support the Completed I/O subtest.

2.5 Mapped I/O

The Mapped I/O subtest is designed to perform synchronous reads/writes operations identical to the Synchronous I/O subtest. The difference is how the transfer buffer is allocated. This subtest will allocate a buffer that is also mapped in kernel space, thereby removing the copy operation to/from user space, ideally increasing I/O operations per second.

At the time of this writing, lxPatIO does not yet support the Mapped I/O subtest.

2.6 CopyFile

The CopyFile subtest is designed to copy files between network clients on the same subnet. In other words, this is supposed to stress the network interface via a "sendfile" type of approach. The data should not have to be copied into user space until the time comes to compare the results.

At the time of this writing, lxPatIO does not yet support the CopyFile subtest.

3 Parameters

This section describes in detail the user-definable parameters available for lxPatIO. Note that while each of them may be user-definable, some may be better left alone. Please read the help for each parameter used in your testing.

3.1 Client password

Client password is the password required to connect to the networking clients through CIFS.

Optional
Default: password
Range: String parameter of up to 80 characters

3.2 Client share

Client share is the share drive to connect to on the networking clients through CIFS.

Optional
Default: c$
Range: String parameter of up to 512 characters

3.3 Client username

Client username is the username used to connect to the networking clients through CIFS.

Optional
Default: Administrator
Range: String parameter of up to 80 characters

3.4 Collect stats

Collect stats specifies whether the test will collect statistics on I/O latency, I/O operations per second and bytes per second. This **does not** affect how the statistics are reported, but rather is just a parameter to specify whether the statistics are collected. Note that this parameter must be enabled if Statistics reporting is enabled.

Optional (but required if Statistics reporting is enabled)
Default: 0
Range: 0 = No statistics collected, 1 = Statistics collected

3.5 Delete old data files

Delete old data files is an indicator that you want to remove all previously created .dat files from the target device before testing commences. If this parameter is off, data files will accumulate on the target. And depending on the Sticky filename count and Save file count parameters, you may fill up the target's storage space.

Optional
Default: 0
Range: 0 = Does not delete old ".dat" files, 1 = Deletes old ".dat" files

3.6 Destination file size

Destination file size is the size of the resultant .dat file residing on the target after all of the write file operations have been performed. This parameter only indirectly affects the performance of the test by allowing more transfers of different data to a target before an open/close is performed. You want to set this to a high value if you are more interested in higher operations-per-second statistics versus "wasting" time opening and closing files.

The one exception is for read-only targets (including targets set by the Read-only parameter). You are not permitted to write to a read-only target, thus the file size on that target is already set. In this case, this parameter will automatically be set to 0.

Required
Default: None (0 for read-only targets)
Range: 1 KB - 4 GB

3.7 Device nodes

Device nodes is a parameter that gives hints to the auto-configuration subtest as to what targets belong to a specific device. This is a parameter modified by module developers and is not meant to be modified by end-users. These hints are discovered and compared within the module code.

Required
Default: As set in the snippet or package file
For developers only - DO NOT MODIFY

3.8 Disable net

Disable net is a parameter checked during auto-configuration only. Network client discovery can be a long process depending on the size of your test network. If you choose to set this parameter, then the client discovery is skipped. Be warned, however, that the resultant configuration will not produce targets for any of the network devices. Thus, no network testing will be performed.

Optional
Default: 0
Range: 0 = Does not disable network client discovery, 1 = Disables network client discovery

3.9 Exclude interface

Exclude interface is a parameter array checked during auto-configuration only. It allows you to specify multiple network interfaces that you do not want lxPatIO to test. You may, for example, have a network interface on your system-under-test that connects to your corporate network, but must not probe that network for possible clients. In this case, you would add that interface to this parameter. This parameter accepts IP addresses. For the Linux version, it also accepts interface names (e.g. eth0).

Optional

Default: No interfaces excluded
Values: IP addresses in the form of A.B.C.D (or interface names, for Linux only)

3.10 Halt on error

Halt on error defines when lxPatIO will terminate testing if an error were to occur. This parameter is used in conjunction with the Retries parameter.

0 = A test thread attempts to restart up to <Retries> times immediately after the error.
1 = A test thread attempts to restart up to <Retries> times at the next pattern change.
2 = Immediately halt all threads upon detecting an error on any test thread.

Optional
Default: 0
Range: 0 - 2 as noted above

3.11 Ignore boot drive

Ignore boot drive will disable all testing against the boot storage device. This includes the entire physical media associated with the boot device. The purpose is to limit stress testing against the boot drive if boot drive longevity is a concern. This parameter is observed only during auto/reconfiguration.

Optional
Default: 0
Range: 0 = Do not ignore the boot drive for testing, 1 = Ignore testing the boot drive

3.12 Interface range

Interface range allows you to specify a range of IP addresses that you want to perform client discovery on for a particular network interface. Only one range per interface is allowed. Interfaces that are not specified will use the default range (all possible IP addresses within the subnet). The syntax of the range field is OS-dependent.

For Linux, it requires the syntax that the program "nmap" supports:

Name	Range
eth0	192.168.0.1/24
eth1	10.7.31.14-228
eth2	12.0-255.0-255.0-255

You can also specify a range to apply to all interfaces by putting the wildcard character "*" in the name field. The range, though, must be in the form of a classless subnet. For example, "/24". However, specifying a range for all interfaces using the wildcard should be used carefully. The test module checks the ranges in the order they are listed. Thus, specifying a wildcard range means all other ranges listed below it are ignored. All ranges specified before it take precedence.

For Windows, TBD:

Name	Range

Optional

Default: Name = *, Range = /24

3.13 Invert

Invert specifies whether you want to perform an inverted pattern test. An inverted pattern test is an additional test performed per pattern where the original "gold" buffer is filled with pattern data that is bit-flipped from the original pattern. Testing against the inverted pattern is performed during the same iteration as the regular pattern test. Thus, your test sequence will look like this...

Pattern0, **~Pattern0**, Pattern1, **~Pattern1**, ..., Pattern(N-1), **~Pattern(N-1)**

...where the inverted pattern tests are represented in bold.

Optional
Default: 0
Range: 0 = Does not perform inverted pattern test, 1 = Performs inverted pattern test

3.14 Iterations

Iterations is a loop limiter that you can specify if you do not specify a Runtime value. One iteration is equivalent to writing (if not Read-only), reading and comparing the full Destination file size. One iteration typically finishes quite quickly, so it's advisable to perform test while modifying this number. Note, however, that the Runtime variable overrides Iterations. So you will need to remove the Runtime variable from the scope of the test in order to take advantage of Iterations.

Optional
Default: 0
Range: 0 - 4294967295 (2^{32} - 1)

3.15 Maximum client count

Maximum client count limits how many client targets per network interface you are planning on testing. This is a parameter that is observed during the client discovery process of auto/reconfiguration. After auto/reconfiguration completes, the Targets parameter for each interface will contain only a number of entries equal to this parameter. However, setting this parameter to 0 disables this parameter so that there is no limit to the networking targets.

Optional
Default: 0 (Disabled)
Range: 0 - (2^{32} - 1) (This upper limit covers all IPv4 addresses)

3.16 Mount root

Mount root is an absolute path that specifies where the user would like network targets mounted during auto/reconfiguration.

Optional
Default: /mnt (Linux), TBD (Windows)
Range: String with length 0 - 512 characters. An empty string (0 characters) signifies using the default.

3.17 Offset

Offset determines how many bytes to incrementally progress through a pattern file for each iteration of testing. For example, if you have a pattern file with contents "0123456789" and Offset of 2, then the first iteration will compare the original data set as is, the second iteration will compare against "2345678901", the third iteration will compare against "4567890123", etc. Eventually, this will loop back to somewhere around the beginning of the pattern, but not necessarily at byte[0]. For example, if you have the same pattern file and an offset of 3, the iterations will be as such: 0123456789, 3456789012, 6789012345, 9012345678, 2345678901, etc.

The Offset parameter can be used regardless if Runtime or Iterations is set. Just note that an increment into the pattern file by an Offset amount counts as one iteration.

Optional
Default: 0 (No incremental offset. The comparison will always start at byte[0] of the pattern)
Range: 0 - (2^{32} - 1) (However, this value will be modded against each pattern file size during testing)

3.18 Outstanding IO ops

Outstanding IO ops is valid only for the Asynchronous I/O and Completed I/O subtests.

Outstanding IO ops determines how many read() and write() operations are allowed per thread assigned to a target (as defined by Thread values).

Optional
Default: 0 ("Disabling" the allowance of more than one I/O operation)
Range: 0 - 256 (The upper limit may be reduced at runtime. The formula is min(256, Destination file size/Transfer size))

3.19 Pattern directory

Pattern directory specifies the directory where the local pattern files are installed. This is a required parameter as no comparisons can be made without patterns.

Required
Default: Linux: /LinuxVSS/patterns, Windows: TBD
Range: Absolute path with maximum of 512 characters

3.20 Patterns

Patterns is an array parameter where you can specify multiple pattern files (ending in .pat) for testing against specified Targets. If there are no slashes in the specified pattern name, then it is assumed this is a pattern file contained within the Pattern directory. Otherwise, the entry will be taken as a full path to the pattern file.

Required
Default: Some patterns may be specified by default depending on the type of test being performed.
Range: Filename(s) with a maximum of 512 characters each.

3.21 R-W completions

R-W completions specifies the allowable count of read/write operations per Transfer size. The read/ write operations are not guaranteed to complete the transfer of the full Transfer size in one call, especially in high-stress testing of the system. With this parameter, you may limit how many completions are acceptable before signaling an error for the test.

Optional
Default: 1
Range: 0 - 255

3.22 Random pattern seed

Random pattern seed is the beginning seed the user may specify when enabling the Randomize parameter. This parameter is ignored if Randomize is disabled. This seed is used for the randomization of the pattern file order. By default, the order of the patterns tested against is the same list as the Patterns parameter. If you do not specify a seed, and Randomize is enabled, a seed will be generated for the test based off of the current time.

Optional
Default: None
Range: 0 - (2^{32} - 1)

3.23 Randomize

Randomize will enable sorting the Patterns list in a random order. If no Random pattern seed is specified, a seed will automatically be generated based off of the current time. The seed, in either case, will be reported out so the user knows what seed was used if they need to repeat the same test.

Optional
Default: 0 (Disabled)
Range: 0/1 (false/true)

3.24 Read-only

Read-only tells the program to treat targets as read-only targets. This means no files will be written to the target. Instead, it is expected that the target already has the pattern files available. If the target does not have the pattern files available, an error will be signaled. If the target does contain the pattern files, then the test commences with read operations only.

Optional/Required: Optional for targets that are read-write-capable. Required for targets that are readonly
Default: Optical drive targets (CD/DVD) = 1. Other drive targets = 0.
Range: 0/1 (false/true)

3.25 Retries

Retries is a parameter that works in conjunction with Halt on error. It serves as a limit to the number of retries allowed by a thread/iteration based off of the specified limiter in the Halt on error parameter. If the read/write/comparison operations exceed this limit, an error is recorded.

Optional
Default: 0 (Disabled)
Range: 0 - 255

3.26 Runtime

Runtime specifies the number of seconds this test should run. The module will divide this amount equally amongst the Patterns list. However, due to setup processes and other system variables that can consume time, each pattern may not be allotted the full section of time for testing. This module accounts for that extra process time and will adjust accordingly. In other words, this module should finish in Runtime seconds +/- 1 second. The exception is if the specified Runtime is very small. This module runs on the premise that each pattern is tested against at least once. So for the minimum value, a good rule-of-thumb is to provide at least 3 seconds per pattern.

Optional (but highly encouraged)
Default: 3600 seconds (1 hour)
Range: 0 - (2^{32} - 1)

3.27 Save file count

Save file count specifies how many test files lxPatIO should save during a test before it starts deleting the older test files. Each test file pertains to a pattern (by default), so when the pattern file changes, a different file is created. The old file is deleted unless the number of files created is lower than this parameter. Once it reaches this limit, it starts deleting the data files, starting with the oldest. Keep in mind that if Sticky filename count is set, you may not get as many saved test files as specified in this parameter.

Optional
Default: 0 (No files saved)
Range: 0 - 1023

3.28 Statistics reporting

Statistics reporting specifies the frequency at which statistics are reported via the test executor. This only affects how often the statistics are reported, not how often they are collected. This parameter is meaningless unless the Collect stats parameter is set appropriately.

Optional
Default: 0 (Disabled)
Range: 0 = Disabled, -1 = Report after pattern change, -2 = Report at end of test, 1 - "Runtime - 1" = Report statistics once every N seconds.

3.29 Sticky filename count

Sticky filename count specifies how many pattern files the test must run through before a test file is renamed.

Optional
Default: 1 (Change test filename after each pattern)
Range: 0 = Use the same filename for the entire test, 1 - 1024 = Change the test filename after N patterns are tested.

3.30 Target directory

Target directory specifies the relative path from the target mount point where the test files will be written to and read from. This is a string with a 512 character limit

Optional
Default: test_files
Range: 512-character limit string

3.31 Targets

Targets is a parameter that is created after auto-configuration is run on a fresh package. It is created automatically upon discovery of targets for a specific device. If this parameter is not present, then either the system has not gone through an auto/reconfiguration, or there are no targets that match that device type. It is highly advisable that the user let lxPatIO create this parameter automatically instead of creating it on their own.

Required (in order for a device type to run tests)
Default: No targets defined for device type

Range: 512-character limit string

3.32 Thread values

Thread values is an array parameter that allows a user to specify, essentially, the number of threads they would like created per target. The column "per drive" is the first multiplier and simply signifies how many test files are created on the target per pattern. The column "per file" is the second multiplier and signifies how many threads should read/write each test file. Multiplying these numbers by the number of targets will yield the thread count per pattern. The main purpose is to stress the target with read/ write operations.

As of this writing, the "per file" value has yet to be implemented in ILVSS. Currently, it will only support one thread per test file.

Optional
Default: 1, 1 (One test file per drive, one thread per test file) **Range:**
(For either column) 1 - 256

3.33 Transfer size

Transfer size specifies the amount of data (in bytes) to transfer to/from the target with each read/write operation. These operations may not successfully transfer the entire amount, but rather only a partial amount. In that case, you can provide a limit to the number of required operations to complete the full transfer size with the R-W completions parameter. This parameter may be larger than the size of any pattern file, but it must be smaller than the Destination file size.

Optional
Default: 128 KB
Range: 8 B - 1 MB

3.34 W-R ratio

W-R ratio specifies how many writes and reads to perform before a comparison is performed. The purpose of any test is to write the full test file, Transfer size bytes at a time, then read the test file, again at Transfer size bytes at a time, then do the comparison. We are comparing the full buffer of Destination file size in bytes. But you have the option of writing the same file multiple times before a read is performed. Likewise, you have the option of performing multiple reads before a comparison is performed.

In addition, you may specify multiple lines of write and read counts. Each line will be used for a single comparison (iteration). It will continuously loop through these values if there are more iterations than ratios to use.

Optional
Default: 1, 1 (One write and one read operation per comparison)
Range: (For either column) 1 - (2^{32} - 1)

3.35 Additional Client shares

Additional client shares allows adding more networking clients through CIFS. Each share added to the array parameter will be added to the target if it can be mounted and has read/write permissions. Add additional shares by specifying share “Name”, “User” and ‘Password”. Add one share per row.

Optional
Default: n/a
Range: String parameter of up to 512 characters