Отчеты о результатах тестирования: справочное руководство

Версия	Дата изменения	Описание изменения	Автор
1.0	28.12.2006	Первая версия документа.	Хорошилов А.В.

Введение

Настоящий документ описывает отчеты о результатах тестирования, выполненного при помощи тестового набора OLVER [1]. Целью этого тестового набора является проверка требований стандарта *Linux Standard Base Core Specification 3.1* [2] относительно функций прикладного бинарного интерфейса, описанных в разделах *III Base Libraries* и *IV Utility Libraries*. Вышеозначенные разделы стандарта определяют требования к наличию и функциональности 1532 функций прикладного бинарного интерфейса операционной системы Linux. В тестовом наборе OLVER все тестируемые функции распределены по *подсистемам* – группам функций, объединенным общей функциональностью. Детали распределения представлены на сайте [1].

В документе приняты следующие соглашения об оформлении текста. Имена директорий, файлов, а также идентификаторы выделяются посредством `моноширинного шрифта`. Части имен файлов и директорий, которые изменяются в зависимости от некоторых условий, выделяются при помощи угловых скобок (например, `<scenario_name>.utt`).

Исполнение тестового набора

В процессе выполнения тестового набора OLVER происходит последовательный запуск тестовых сценариев, перечисленных в файле `/opt/olver/testplan`. Каждый тестовый сценарий предназначен для тестирования группы функций, связанных общей функциональностью.

Информация о ходе работы каждого тестового сценария сохраняется в отдельном файле, называемом *трассой теста*. Этот файл имеет имя `<scenario_name>.utt` и располагается в каталоге `/var/opt/olver/<TIMESTAMP>`, где `<TIMESTAMP>` – время запуска тестового набора в формате "ГОД-МЕСЯЦ-ДЕНЬ_ЧАС-МИНУТА-СЕКУНДА".

После завершения процесса тестирования происходит генерация набора отчетов о результатах тестирования. В состав набора входят следующие отчеты:

- суммарный отчет о результатах тестирования;
- детальный отчет о результатах тестирования;
  - отчет об обнаруженных ошибках;
  - отчет о результатах работы тестовых сценариев;
  - отчет о покрытии ветвей функциональности;
- отчет о покрытии требований стандарта.

Ниже представлена информация о содержимом этих отчетов.

Суммарный отчет

Суммарный отчет о результатах тестирования генерируется в файле `/var/opt/olver/<TIMESTAMP>/summary.htm`. Отчет состоит из трех частей: заголовка, списка обнаруженных проблем и списка протестированных функций.

Заголовок содержит статистическую информацию о ходе выполнения тестов.

Summary Report for Test Run

Host 'kpm' on OS 'Linux 2.6.9-42.ELsmp' at Time 2006/12/25-02:04:48

Scenarios		Problems		Requirements	
Passed	243	Known	105	Checked	6675
Failed	37	New	11	Failed	262
Unresolved	0	Internal	1	Total	6937
Total	280	Total	117	Checked	
Details...		Details...		Details...	

Tested Functions: 1530. Details...

В первой строке заголока указаны имя машины, на которой выполнялось тестирование, идентификатор операционной системы, а также время запуска тестов. Следующая за ней таблица описывает количественые характеристики, выполненного запуска тестового набора.

В столбеце ‘Scenarios’ представлена следующая информация:

- Passed – число тестовых сценариев, не обнаруживших проблем;
- Failed – число тестовых сценариев, обнаруживших проблемы;
- Unresolved – число тестовых сценариев, статус завершения которых неизвестен;
- Total – общее число тестовых сценариев, участвовавших в данном запуске тестового набора.

Столбец ‘Problems’ содержит данные об обнаруженных проблемах.

- Known – число проблемных ситуаций, идентифицированных как уже известные разработчикам тестов ошибки в тестируемой системе;
- New – число обнаруженных несоответствий между поведением тестируемой системы и требованиями стандарта, но не идентифицированных как уже известные ошибки в тестируемой системе;
- Internal – число выявленных проблем, которые могут быть вызваны как ошибками в самом тестовом наборе, так и проблемами в тестируемой системе;
- Total – общее число обнаруженных проблем.

Столбец ‘Requirements’ описывает статистику проверки требований стандарта.

- Checked – число элементарных требований стандарта, успешно проверенных в ходе данного запуска тестового набора;
- Failed – число элементарных требований стандарта, нарушение которых было выявлено в ходе данного запуска тестового набора;
- Total checked – общее число элементарных требований стандарта, проверявшихся в ходе данного запуска тестового набора.

В заключении заголовка указано число целевых функций, протестированных в ходе данного запуска тестового набора.

Вслед за заголовком суммарного отчета представлена более детальная информация об обнаруженных проблемах. Для каждого вида проблем (Known, New и Internal) в отчете присутствует отдельная таблица. Таблица Known Problems

Known Problems

Scenario	Failure Id	Bug Id	Description
attr_scenario	failure 1	bug526_1 (pthread_attr_setstack)	bug526_1(pthread_attr_setstack) (pthread_create_cancel_scenario.utt) Function pthread_attr_setstack is not implemented.
chgat_scenario	failure 2	bug329(chgat)	bug329(chgat) (trace chgat_scenario.utt) If the function is invoked not right after addstr(), it doesn't work.
conversion_scenario	failure 3	bug380(asctime)	bug380(asctime) (conversion_scenario.utt) If the function is unsuccessful, it should return NULL, but returns a string, for example "??? Oct 9 09:09:09 1909"
conversion_scenario	failure 4	bug379(asctime_r)	bug379(asctime_r) (conversion_scenario.utt) If the function is unsuccessful, it should return NULL, but returns a string, for example "??? Oct 9 09:09:09 1909"
conversion_scenario	failure 6	bug376(ctime)	bug376(ctime) (conversion_scenario.utt) The difference between local and global time is equal to 2.5, while timezone has been set to -10800 (3 hours).

содержит 4 столбца:

- Scenario – имя тестового сценария, обнаружевшего проблему;
- Failure Id – идентификатор проблемы, в рамках данного запуска тестового набора, и гиперссылка на более детальную информацию о данной проблеме;
- Bug Id – идентификатор известной ошибки в тестируемой системе;
- Description – краткое описание данной ошибки.

Таблица New Problems описывает несоответствия между поведением тестируемой системы и требованиями стандарта, не идентифицированные как уже известные ошибки в тестируемой системе.

New Problems

Scenario	Failure Id	Req Id	Description
float_scenario	failure 30	logbf.04	Postcondition failed Requirement failed: {logbf.04} It shall return the exponent of x
float_scenario	failure 122	scalblnl.10	Postcondition failed Requirement failed: {scalblnl.10} If the correct value would cause underflow, and is representable, the correct value shall be returned
math_exp_scenario	failure 229	logf.04	Postcondition failed Requirement failed: {logf.04} functions shall return the natural logarithm of x
math_exp_scenario	failure 257	powl.05	Postcondition failed Requirement failed: {powl.05} finite values of x < 0, and finite non-integer values of y, NaN shall be returned
math_exp_scenario	failure 269	powl.04	Postcondition failed Requirement failed: {powl.04} functions shall return the value of x raised to the power y

В этой таблице содержится 4 столбца:

- Scenario – имя тестового сценария, обнаружевшего проблему;
- Failure Id – идентификатор проблемы, в рамках данного запуска тестового набора, и гиперссылка на более детальную информацию о данной проблеме;
- Req Id – идентификатор атомарного требования стандарта, нарушение которого является причиной появления данной проблемы;
- Description – краткий текст нарушенного требования стандарта.

Таблица Internal Problems описывает проблемы, которые могут быть вызваны как ошибками в самом тестовом наборе, так и проблемами в тестируемой системе.

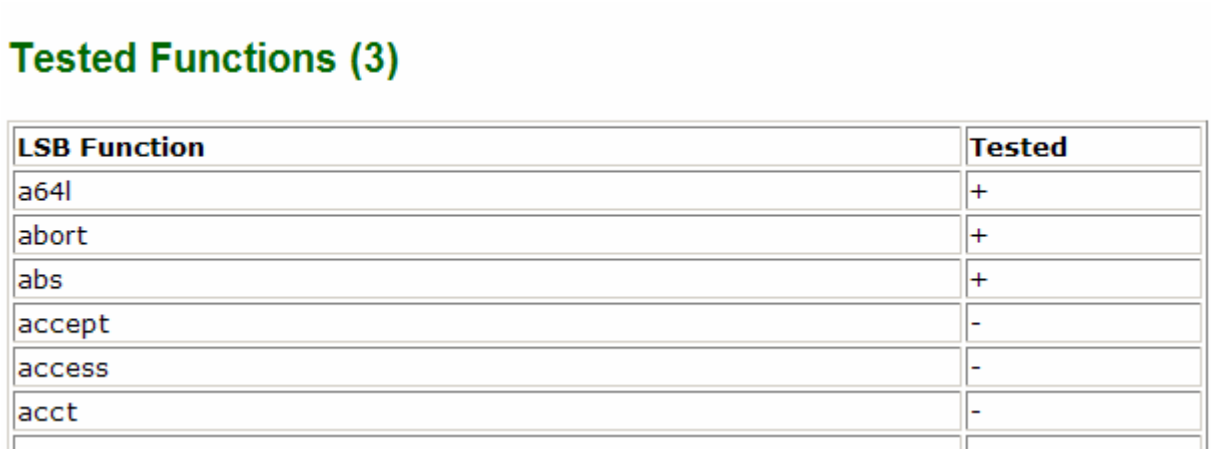
Internal Problems

Scenario	Failure Id	Failure Description
float_scenario	failure 136	Mediator failed Function not found in any library: __signbit

В этой таблице содержится 3 столбца:

- Scenario – имя тестового сценария, обнаружевшего проблему;
- Failure Id – идентификатор проблемы, в рамках данного запуска тестового набора, и гиперссылка на более детальную информацию о данной проблеме;
- Failure Description – краткое описание проблемы.

Суммарный отчет завершается списком функций стандарта LSB Core Generic 3.1 с выделением тех функций, которые были протестированы в ходе данного запуска тестового набора.



Tested Functions (3)

LSB Function	Tested
a64l	+
abort	+
abs	+
accept	-
access	-
acct	-

Детальный отчет о результатах тестирования

Детальный отчет о результатах тестирования генерируется в файле `/var/opt/olver/<TIMESTAMP>/report/index.html`, а также доступен по ссылке `Details…` из столбца ‘Scenarios’ заголовка суммарного отчета. Детальный отчет состоит из множества HTML-сраниц, связанных между собой гиперссылками. Для навигации по страницам служит меню, расположенное слева страницы. Меню навигации состоит из трех разделов: ‘Failures’, ‘Scenarios’ и ‘Stimuli & reactions’.

Failures

All failures (1045)

Grouped failures

Scenarios

All scenarios

_Exit_scenario
__cxa_atexit_scenario
__libc_start_main_scenario
wstrint_scenario
wstrreal_scenario
wtoken_scenario

Stimuli & reactions

All stimuli & reactions

fs.dir
fs.fifo
fs.fs
fs.ftw
fs.glob
fs.meta
fs.name

Раздел ‘Failures’ ведет к различным представлениям обнаруженных проблем. По ссылке ‘All failures’ проблемы представлены в виде таблицы, а по ссылке ‘Grouped failures’ – в виде дерева, в котором проблемы сгруппированы согласно принципам, описанным ниже.

Раздел ‘Scenarios’ содержит ссылку на отчет о результатах работы всех тестовых сценариев, а также ссылки на более детальную информацию о результатах работы каждого отдельного сценария.

Раздел ‘Stimuli & reactions’ позволяет анализировать отчеты о покрытии ветвей функциональности тестируемых функций. Ссылка ‘All stimuli & reactions’ ведет к суммарному отчету о покрытии, а последующий набор ссылок – к отчетам о покрытии по каждой подсистемы в отдельности.

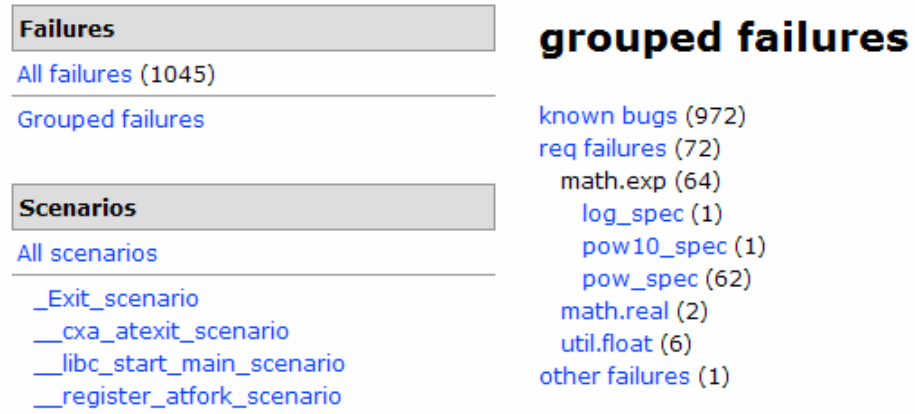
Отчет об обнаруженных проблемах

Представление ‘All failures’ отчета об обнаруженных проблемах содержит таблицу с полным списком проблем, их кратким описаниями и ссылками на детальный отчет, посвященный каждой проблеме в отдельности.

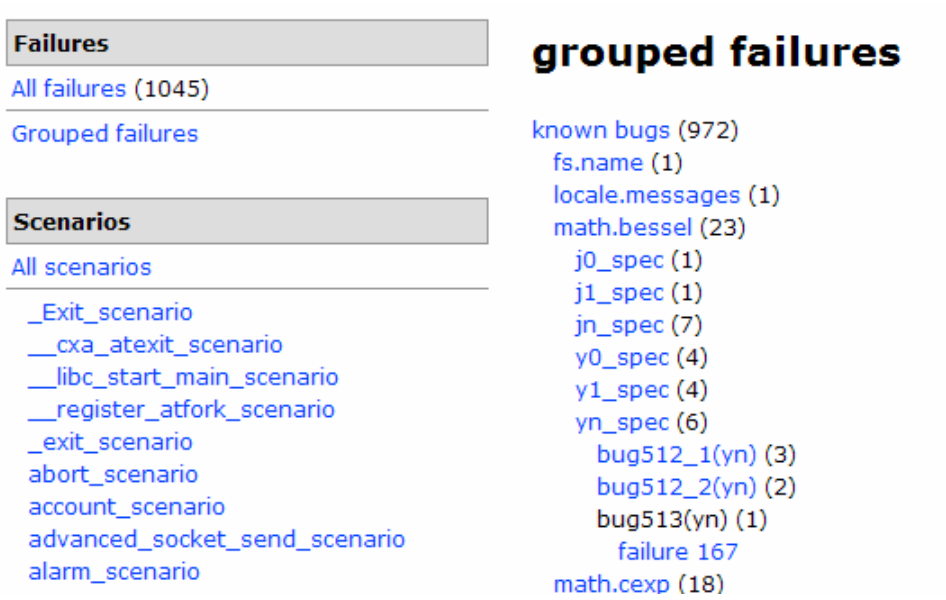
Второе представление ‘Grouped failures’ содержит группировку проблем согласно следующим принципам. Все обнаруженные проблемы изначально делятся на три класса:

- known bugs – проблемы, идентифицированные как уже известные ошибки в тестируемой системе;
- req failures – проблемы, заключающиеся в несоответствии между поведением тестируемой системы и требованиями стандарта, но не идентифицированных как уже известные ошибки в тестируемой системе;
- other failures – проблемы, которые могут быть вызваны как ошибками в самом тестовом наборе, так и проблемами в тестируемой системе.

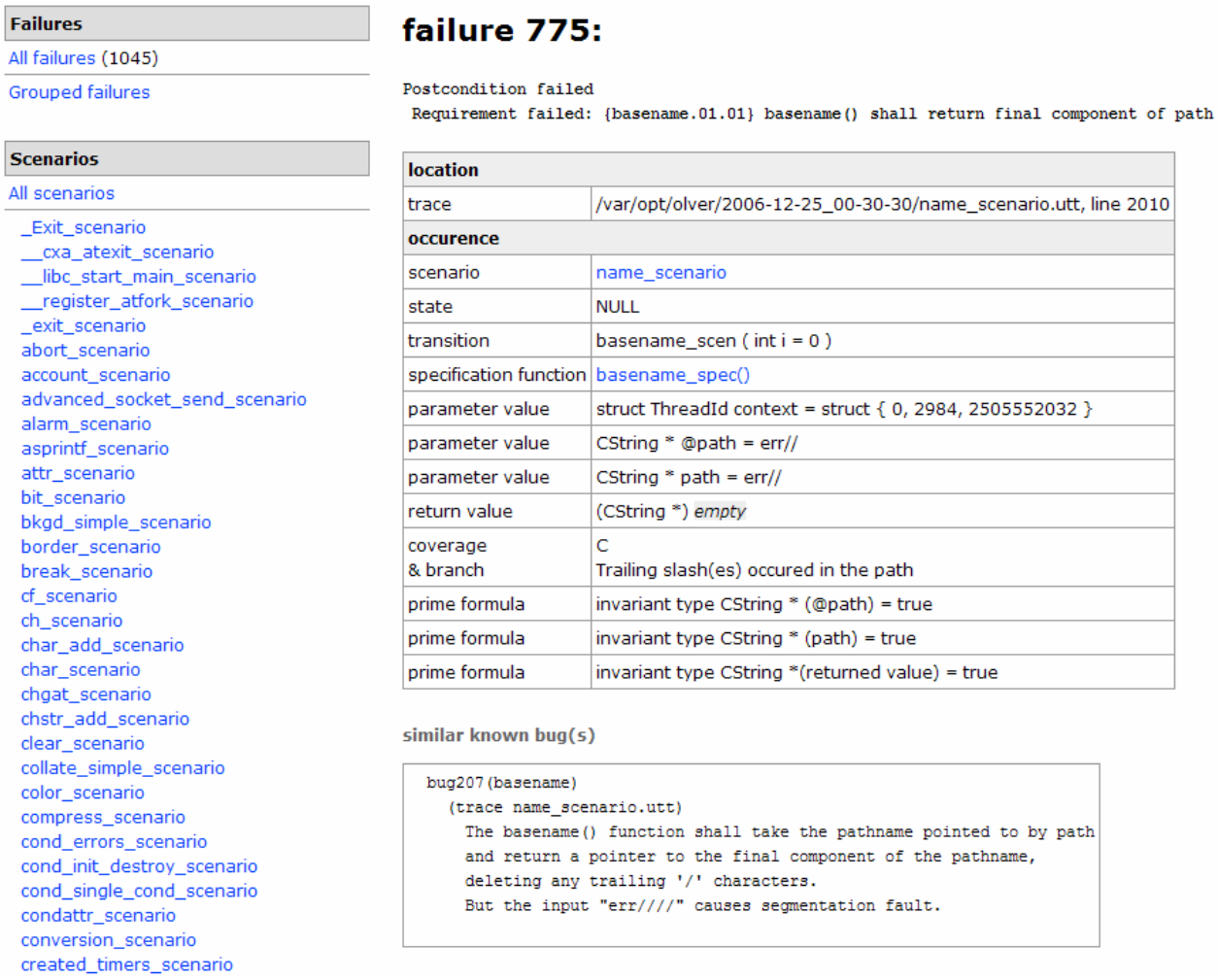
Следующие два уровня группировки являются общими для всех трех классов проблем – это уровени группировки по подсистемам и по отдельным функциям.



Четвертый уровень группировки присутствует только в классе ‘known bugs’ – это уровень группировки по известным ошибкам.

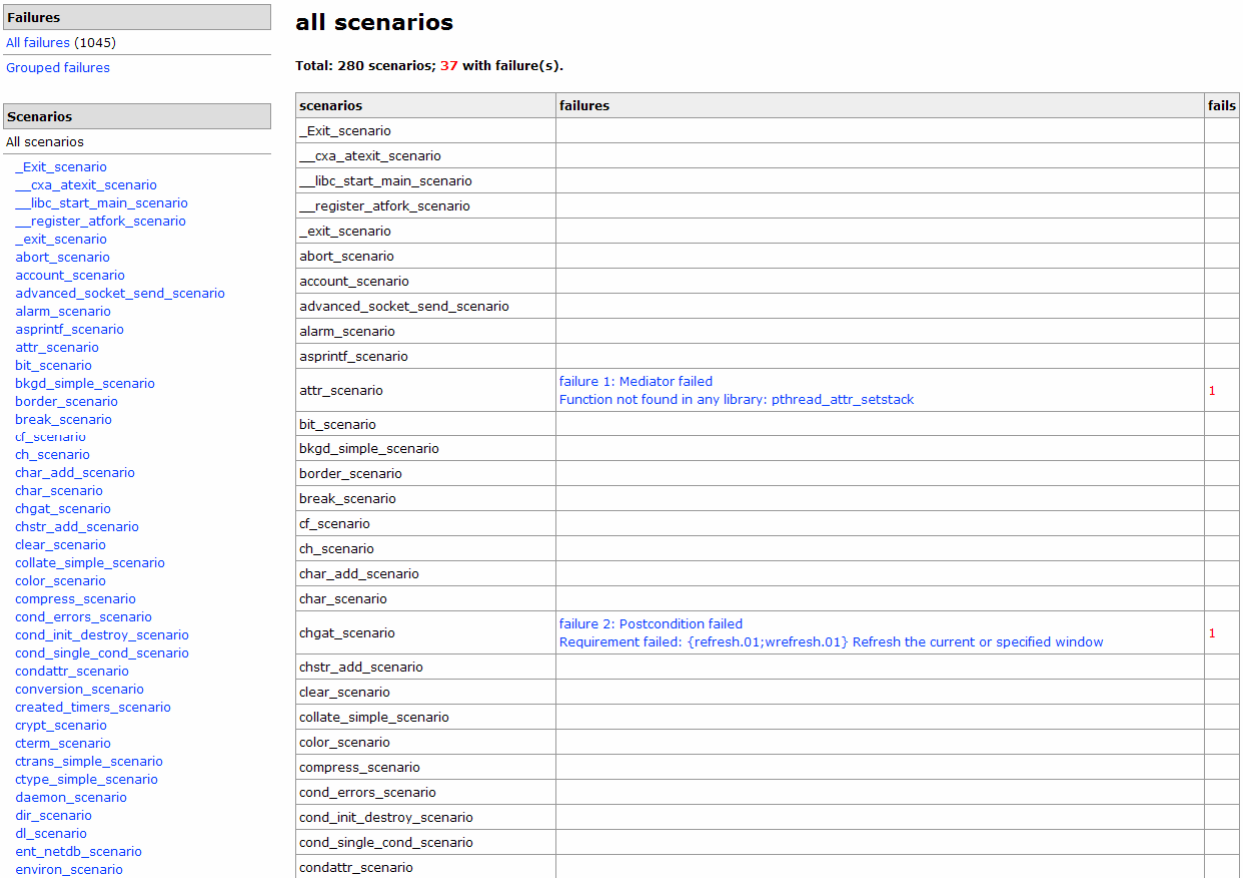


В качестве листовых вершин дерева проблем содержатся ссылки на детальные отчеты, посвященные каждой проблеме в отдельности. Детальное отчет по проблеме содержит краткое описание проблемы, информацию о значениях параметров функции и описание уже известной ошибки, соответствующей данной проблеме, если таковая была обнаружена.



Отчет о результатах работы тестовых сценариев

Первая строка отчета о результатах работы всех тестовых сценариев (ссылка ‘All scenarios’) содержит общее число тестовых сценариев, участвовавших в подотчетном запуске тестового набора, а также число сценариев, обнаружевших проблемы. Вторым элементом отчета является таблица, в которой устанавливается соответствие между тестовыми сценариями и обнаруженными проблемами.



Failures

All failures (1045)

Grouped failures

Scenarios

All scenarios

_Exit_scenario
__cxa_atexit_scenario
__libc_start_main_scenario
__register_atfork_scenario
_exit_scenario
abort_scenario
account_scenario
advanced_socket_send_scenario
alarm_scenario
asprintf_scenario
attr_scenario
bit_scenario
bkgd_simple_scenario
border_scenario
break_scenario
cf_scenario
ch_scenario
char_add_scenario
char_scenario
chgat_scenario
chstr_add_scenario
clear_scenario
collate_simple_scenario
color_scenario
compress_scenario
cond_errors_scenario
cond_init_destroy_scenario
cond_single_cond_scenario
condattr_scenario
conversion_scenario
created_timers_scenario
crypt_scenario
cterm_scenario
ctrans_simple_scenario
ctype_simple_scenario
daemon_scenario
dir_scenario
dl_scenario
ent_netdb_scenario
environ_scenario

all scenarios

Total: 280 scenarios; 37 with failure(s).

scenarios	failures	fails
_Exit_scenario		
__cxa_atexit_scenario		
__libc_start_main_scenario		
__register_atfork_scenario		
_exit_scenario		
abort_scenario		
account_scenario		
advanced_socket_send_scenario		
alarm_scenario		
asprintf_scenario		
attr_scenario	failure 1: Mediator failed Function not found in any library: pthread_attr_setstack	1
bit_scenario		
bkgd_simple_scenario		
border_scenario		
break_scenario		
cf_scenario		
ch_scenario		
char_add_scenario		
char_scenario		
chgat_scenario	failure 2: Postcondition failed Requirement failed: {refresh.01;wrefresh.01} Refresh the current or specified window	1
chstr_add_scenario		
clear_scenario		
collate_simple_scenario		
color_scenario		
compress_scenario		
cond_errors_scenario		
cond_init_destroy_scenario		
cond_single_cond_scenario		
condattr_scenario		

Пример детального отчета о результатах работы отдельного тестового сценария приведен на следующем рисунке:

Failures

All failures (1045)

Grouped failures

Scenarios

All scenarios

_Exit_scenario
__cxa_atexit_scenario
__libc_start_main_scenario
__register_atfork_scenario
_exit_scenario
abort_scenario
account_scenario
advanced_socket_send_scenario
alarm_scenario
asprintf_scenario
attr_scenario
bit_scenario
bkgd_simple_scenario
border_scenario
break_scenario
cf_scenario
ch_scenario
char_add_scenario
char_scenario
chgat_scenario
chstr_add_scenario
clear_scenario
collate_simple_scenario
color_scenario

scenario wstrreal_scenario

execution
trace: /var/opt/olver/2006-12-25_00-30-30/wstrreal_scenario.utt start: Mon Dec 25 02:04:47 MSK 2006 end: Mon Dec 25 02:04:47 MSK 2006 Product Name: CTesK Product Build: 20061127 Host: kpm Product Version: 2.2.5 Operating System: Linux 2.6.9-42.ELsmp

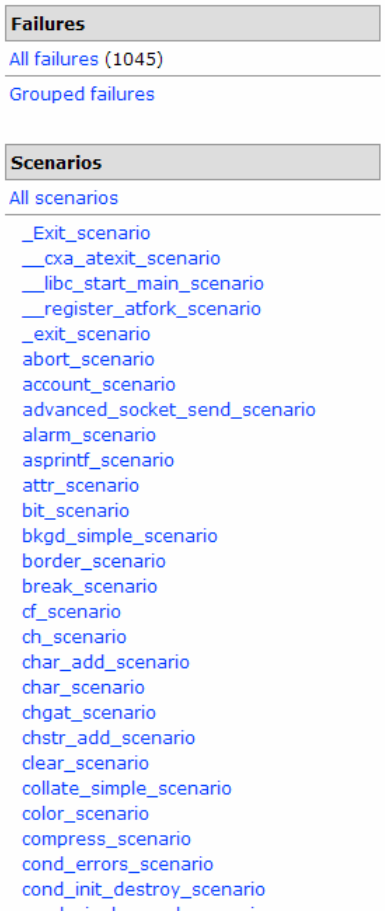
failures	fails
failure 1040: Postcondition failed Requirement failed: {__wcstod_internal.wcstod.17} If the correct value would cause underflow, the smallest normalized positive number shall be returned failure 1041: Postcondition failed Requirement failed: {wcstod.17} If the correct value would cause underflow, the smallest normalized positive number shall be returned failure 1042: Postcondition failed Requirement failed: {__wcstof_internal.wcstof.17} If the correct value would cause underflow, the smallest normalized positive number shall be returned failure 1043: Postcondition failed Requirement failed: {wcstof.17} If the correct value would cause underflow, the smallest normalized positive number shall be returned failure 1044: Postcondition failed Requirement failed: {__wcstold_internal.wcstold.17} If the correct value would cause underflow, the smallest normalized positive number shall be returned failure 1045: Postcondition failed Requirement failed: {wcstold.17} If the correct value would cause underflow, the smallest normalized positive number shall be returned	6

Этот отчет содержит информацию о среде и времени выполнения тестового сценария, а также список всех проблемных ситуаций, обнаруженных этим сценарием.

Отчет о покрытии ветвей функциональности

Технология тестирования UniTesK [3], на которой базируется тестовый набор OLVER, поддерживает механизм оценки качества тестирования на основе разбиения пространства входных значений спецификационных функций на классы эквивалентности. Эти классы эквивалентности также называются ветвями функциональности тестируемых функций. Для одной функции можно определить несколько разбиений на классы эквивалентности, выделяя ветви функциональности исходя из различных принципов. Подробнее механизм оценки качества тестирования на основе ветвей функциональности описан в документе «Тестирование на основе формальных спецификаций: быстрое знакомство» [4] и в материалах сайта [3].

Ссылка ‘All stimuli & reactions’ ведет к суммарному отчету о покрытии ветвей функциональности. В этом отчете для каждой подсистемы указан процент покрытия ветвей функциональности по каждому виду покрытий. В первой колонке отчета

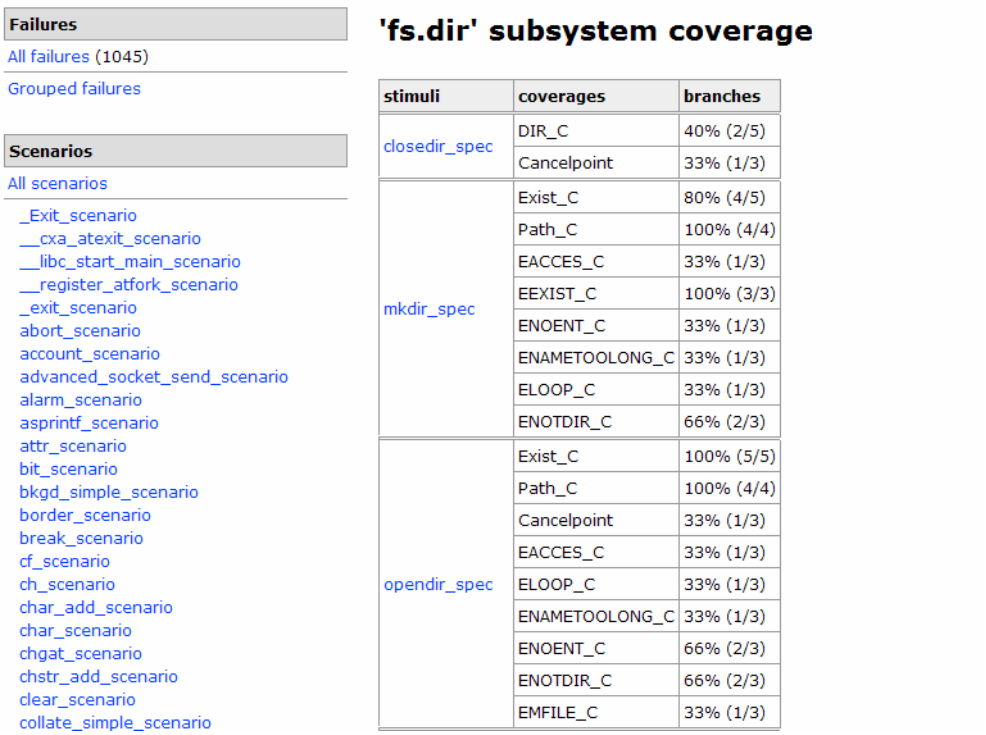


specification functions coverage

subsystems	coverages	branches	fails
fs.dir	Cancelpoint	33% (6/18)	–
	DIR_C	45% (16/35)	
	EACCES_C	33% (3/9)	
	EEXIST_C	100% (6/6)	
	ELOOP_C	33% (2/6)	
	EMFILE_C	33% (1/3)	
	ENAMETOOLONG_C	33% (2/6)	
	ENOENT_C	55% (5/9)	
	ENOTDIR_C	66% (4/6)	
	Exist_C	86% (13/15)	
	Path_C	91% (11/12)	
fs.fifo	C	100% (6/6)	–
fs.fs	C	100% (5/5)	–
fs.ftw	C	50% (4/8)	–
fs.glob	C	34% (16/46)	–
	C_errfunc	75% (6/8)	
fs.meta	C	100% (18/18)	–
fs.name	C	61% (8/13)	1
fs.symlink	C_Bufsize	100% (3/3)	–
	C_Path	100% (6/6)	
	C_Symlink	83% (5/6)	
	Cancelpoint	100% (3/3)	

указывается имя подсистемы, по второй – название разбиения на классы эквивалентности, а в третьей – процент покрытия по данному разбиению и количественные показатели. Последняя колонка содержит дополнительную информацию о числе проблем, обнаруженных в соответствующей подсистеме.

Гиперссылка, связанная с именем подсистемы, ведет к отчету о покрытии этой подсистемы. Этот отчет аналогичен предыдущему, но в нем рассматривается покрытие отдельных функций в рамках выбранной подсистемы. В первой колонке содержится имя функции, по второй – название разбиения на классы эквивалентности, а в третьей – процент покрытия по данному разбиению и количественные показатели. Четвертая колонка показывает число ошибок, обнаруженных в соответствующей функции. Если в



Failures

All failures (1045)

Grouped failures

Scenarios

All scenarios

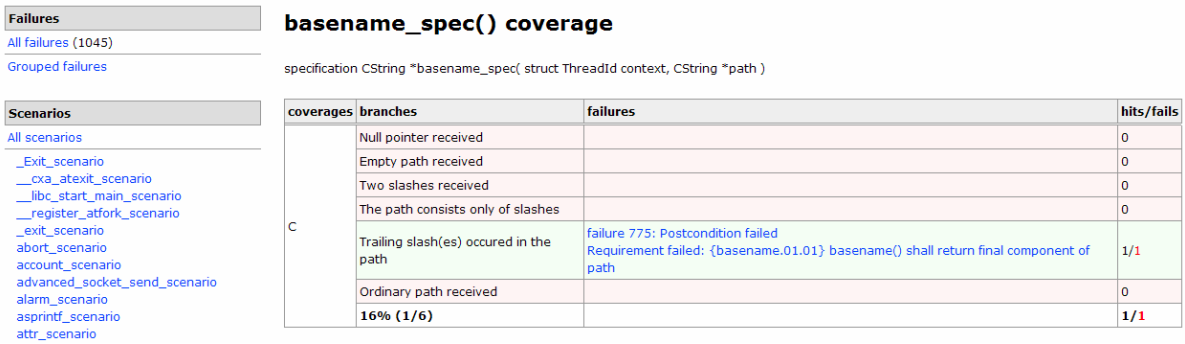
_Exit_scenario
__cxa_atexit_scenario
__libc_start_main_scenario
__register_atfork_scenario
_exit_scenario
abort_scenario
account_scenario
advanced_socket_send_scenario
alarm_scenario
asprintf_scenario
attr_scenario
bit_scenario
bkgd_simple_scenario
border_scenario
break_scenario
cf_scenario
ch_scenario
char_add_scenario
char_scenario
chgat_scenario
chstr_add_scenario
clear_scenario
collate_simple_scenario

'fs.dir' subsystem coverage

stimuli	coverages	branches
closedir_spec	DIR_C	40% (2/5)
	Cancelpoint	33% (1/3)
mkdir_spec	Exist_C	80% (4/5)
	Path_C	100% (4/4)
	EACCES_C	33% (1/3)
	EEXIST_C	100% (3/3)
	ENOENT_C	33% (1/3)
	ENAMETOOLONG_C	33% (1/3)
	ELOOP_C	33% (1/3)
	ENOTDIR_C	66% (2/3)
opendir_spec	Exist_C	100% (5/5)
	Path_C	100% (4/4)
	Cancelpoint	33% (1/3)
	EACCES_C	33% (1/3)
	ELOOP_C	33% (1/3)
	ENAMETOOLONG_C	33% (1/3)
	ENOENT_C	66% (2/3)
	ENOTDIR_C	66% (2/3)
	EMFILE_C	33% (1/3)

выбранной подсистеме не было обнаружено ни одной ошибки, то четвертая колонка отсутствует.

Наиболее детальный уровень отчета о покрытии ветвей функциональности – отчет о покрытии ветвей конкретной функции, доступен по гиперссылке, связанной с именем функции в первой колонке предыдущего отчета.



Failures

All failures (1045)

Grouped failures

Scenarios

All scenarios

_Exit_scenario
__cxa_atexit_scenario
__libc_start_main_scenario
__register_atfork_scenario
_exit_scenario
abort_scenario
account_scenario
advanced_socket_send_scenario
alarm_scenario
asprintf_scenario
attr_scenario

basename_spec() coverage

specification CString *basename_spec(struct ThreadId context, CString *path)

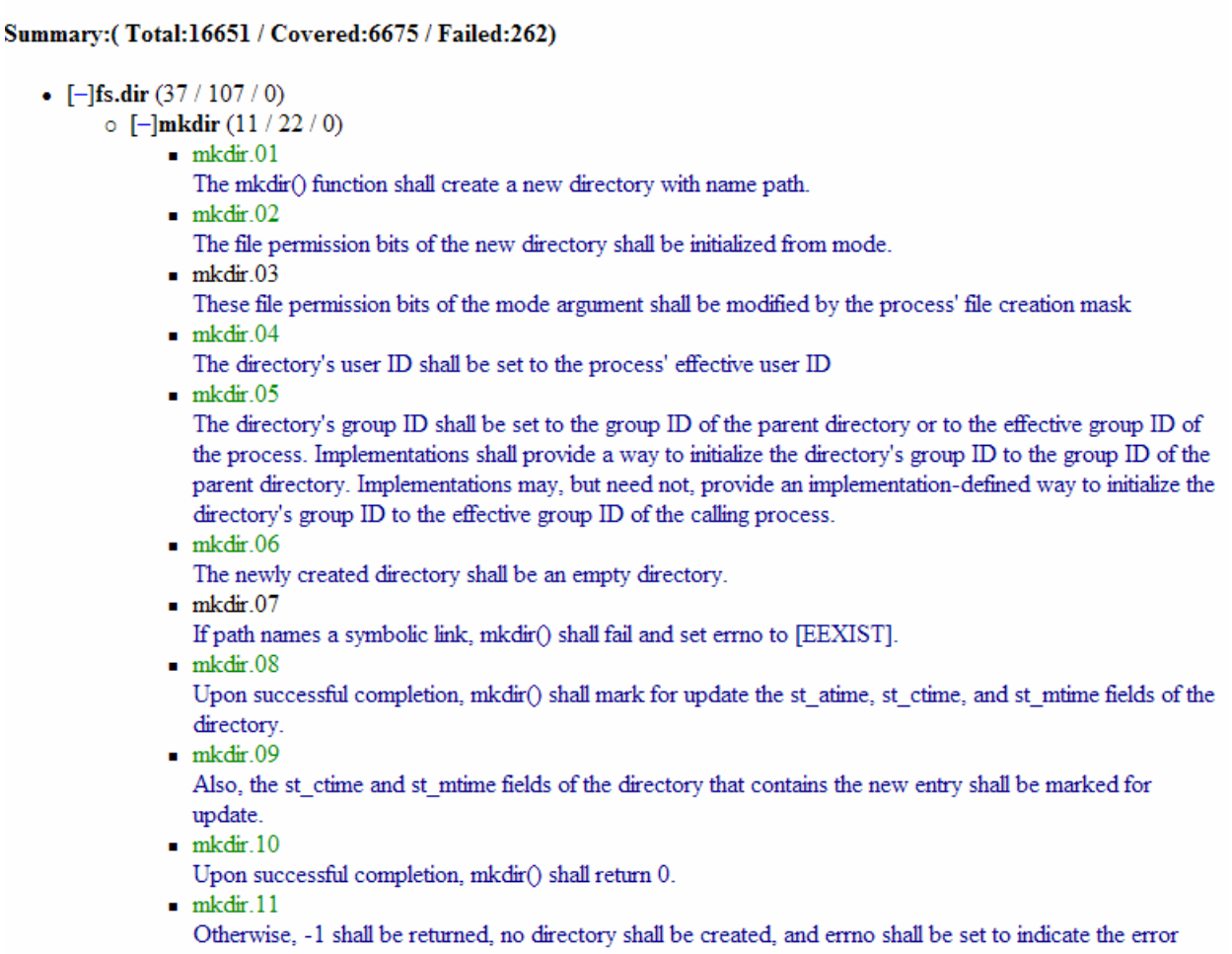
coverages	branches	failures	hits/fails
C	Null pointer received		0
	Empty path received		0
	Two slashes received		0
	The path consists only of slashes		0
	Trailing slash(es) occured in the path	failure 775: Postcondition failed Requirement failed: {basename.01.01} basename() shall return final component of path	1/1
	Ordinary path received		0
	16% (1/6)		**1/1**

В этом отчете описываются все разбиения на ветви функциональности для выбранной функции. В первой колонке таблицы содержатся имена разбиений, а во второй колонке перечислены ветви функциональности, входящие в соответствующее разбиение и процент их покрытия. Строчки таблицы, соответствующие ветвям функциональности, которые были проверены в ходе тестирования, выделены зеленым фоном, а строчки, соответствующие непроверенных ветвям – красным. В последнем столбце таблицы указано число вызовов тестируемой функции, в ситуации связанной с соответствующей ветвью функциональности. Кроме того, если в ходе тестирования был обнаружены проблемы в работе выбранной функции, то в рассматриваемом отчете присутствует дополнительный столбец, описывающий распределение обнаруженных проблем по выделенным ветвям функциональности.

Отчет о покрытии требований стандарта

Отчет о покрытии требований стандарта генерируется в файле `/var/opt/olver/<TIMESTAMP>/result.htm`, а также доступен по ссылке `Details…` из столбца ‘Requirements’ заголовка суммарного отчета.

Первая строка отчета о покрытии требований содержит общую статистику покрытия. Первое число в строке (Total) указывает общее число элементарных требований стандарта LSB Core 3.1 к тестируемым функциям, второе число (Covered) – число проверенных требований, а третье (Failed) – число элементарных требований, нарушение которых было обнаружено тестовым набором в ходе отчетного запуска.



Summary:(Total:16651 / Covered:6675 / Failed:262)

- [–]**fs.dir** (37 / 107 / 0)
  - [–]**mkdir** (11 / 22 / 0)
    - mkdir.01
      The mkdir() function shall create a new directory with name path.
    - mkdir.02
      The file permission bits of the new directory shall be initialized from mode.
    - mkdir.03
      These file permission bits of the mode argument shall be modified by the process' file creation mask
    - mkdir.04
      The directory's user ID shall be set to the process' effective user ID
    - mkdir.05
      The directory's group ID shall be set to the group ID of the parent directory or to the effective group ID of the process. Implementations shall provide a way to initialize the directory's group ID to the group ID of the parent directory. Implementations may, but need not, provide an implementation-defined way to initialize the directory's group ID to the effective group ID of the calling process.
    - mkdir.06
      The newly created directory shall be an empty directory.
    - mkdir.07
      If path names a symbolic link, mkdir() shall fail and set errno to [EEXIST].
    - mkdir.08
      Upon successful completion, mkdir() shall mark for update the st_atime, st_ctime, and st_mtime fields of the directory.
    - mkdir.09
      Also, the st_ctime and st_mtime fields of the directory that contains the new entry shall be marked for update.
    - mkdir.10
      Upon successful completion, mkdir() shall return 0.
    - mkdir.11
      Otherwise, -1 shall be returned, no directory shall be created, and errno shall be set to indicate the error

Далее следует дерево элементарных требований, в котором:

- вершины первого уровня соответствуют подсистемам и содержат статистику покрытия требований стандарта к функциям соответствующей подсистемы в формате (Total / Covered / Failed);
  - вершины второго уровня соответствуют функциям и содержат статистику покрытия требований стандарта к соответствующей функции в том же формате;
    - вершины третьего уровня содержат элементарные требования и в том числе идентификатор требования и текст требования.

Представление дерева позволяет скрывать и открывать все вершины, лежащие ниже любой не листовой вершины. Для этого необходимо кликнуть на знаки [+]/[–], расположенные слева от текста вершины дерева.

Идентификаторы требований, которые не проверялись в ходе отчетного запуска, имеют черный цвет. Идентификаторы требований, которые были проверены в ходе отчетного запуска и для которых проверка не обнаружила нарушений со стороны тестируемй системы, отмечены зеленым цветом. Идентификаторы требований, нарушения которых были выявлены, выделены красным цветом и пометкой (FAILED). Серым цветом обладают идентификаторы непроверяемых требований, таких как требования к приложениям, а не к тестируемой системе.

Рекомендации по использованию отчетов о результатах тестирования

В зависимости от целей, для которых используется тестовый набор OLVER, при анализе результатов тестирования следует обратить внимание на различные виды отчетов. В настоящем разделе представлено несколько рекомендаций по использованию отчетов.

Если тестовый набор используется для **получения информации о соответствии** тестируемой системы требованиям стандарта LSB Core 3.1, то в первую очередь следует проанализировать суммарный отчет о результатах тестирования. Заголовок этого отчета предоставляет статистическую информацию:

- об обнаруженных проблемах (столбец ‘Problems’);
- о числе тестовых сценариев, обнаруживших проблемы (столбец ‘Scenarios’, строка Failed);
- о числе элементарных требований стандарта, нарушение которых было зафиксировано (столбец ‘Requirements’, строка Failed).

Более детальная информация об обнаруженных проблемах также доступна в суммарном отчете о результатах тестирования сразу вслед за заголовком отчета. Кроме того, полезной будет информация о распределении обнаруженных проблем по подсистемам и функциям бинарного интерфейса Linux, которая представлена в разделе ‘Grouped failures’ детального отчета о результатах тестирования.

Если тестовый набор используется для **детального разбирательства с несоответствиями** тестируемой системы требованиям стандарта LSB Core 3.1, то наиболее важно исследовать детальный отчет о результатах тестирования, и, в особенности, раздел, посвященный обнаруженным ошибках. В этом разделе присутствует информация как о распределении проблем по подсистемам и функциям бинарного интерфейса Linux, так и детальные отчеты, посвященные каждой проблеме в отдельности. В дополнение к статическим отчетам, для анализа причин обнаруженных несоответствий будет полезно использовать инструмент динамического воспроизведения трассы теста (Trace Player), входящий в состав набора инструментов CTesK [3].

Если **информация о качестве проведенного тестирования** также является важной, то следует обратить внимание на отчет о покрытии требований стандарта и раздел, посвященный покрытию ветвей функциональности, в детальный отчет о результатах тестирования. Первый отчет позволяет оценить качество тестирование в терминах покрытия элементарных требований стандарта, а второй – в терминах ветвей функциональности отдельных функций. В этих отчетах содержится как статистические данные о покрытии в целом, так и детальная информация о покрытии каждой отдельной функции.

Дополнительная информация

[1] Центр Верификации ОС Linux (http://www.linuxtesting.ru)

[2] Linux Standard Base Core Specification 3.1 (http://refspecs.freestandards.org/LSB_3.1.0/LSB-Core-generic/LSB-Core-generic.html)

[3] Сайт, посвященный технологии UniTesK (http://www.unitesk.com)

[4] Тестирование на основе формальных спецификаций: быстрое знакомство (http://linuxtesting.ru/downloads/olver_demo_getting_started.pdf)