# Stressing Linux with Real-World Workloads

Mark Wong
*Open Source Development Labs*
markw@osdl.org

**Abstract**

Open Source Development Labs (OSDL™) have developed three freely available real-world database workloads to characterize the performance of the Linux kernel. These customizable workloads are modeled after the Transaction Processing Performance Council's industry standard W, C, and H benchmarks, and are intended for kernel developers with or without database management experience. The Scalable Test Platform can be used by those who do not feel they have the resources to run a large scale database workload to collect system statistics and kernel profile data. This paper describes the kind of real world activities simulated by each workload and how they stress the Linux kernel.

## 1 Introduction

OSDL[1] is dedicated to providing developers with resources to build enterprise enhancements into the Linux® kernel and its Open Source solution stacks. This paper focuses on real-world database workloads, based on industry standard benchmarks, used to stress Linux.

OSDL currently provides three workloads, derived from specifications published by the Transaction Processing Performance Council[2] (TPC). TPC is a non-profit corporation created to define database benchmarks with which performance data can be disseminated to the industry.

TPC benchmarks are intended to be used as a competitive tool. All published TPC benchmark results must comply with strict publication rules and auditing to ensure fair comparisons between competitors. Furthermore, it is required that all the hardware and software used in a benchmark must be commercially available with a full disclosure report of the pricing of all the products used as well as support and maintenance costs.

As a basis for our three workloads, we used the TPC Benchmark* W (TPC-W*), TPC Benchmark C (TPC-C*), and TPC Benchmark H (TPC-H*). Each benchmark is briefly described here. TPC-W is a Web commerce benchmark, simulating the activities of Web browsers accessing an on-line book reseller for browsing, searching or ordering. TPC-C is an on-line transaction processing (OLTP) benchmark, simulating the activities of a supplier managing warehouse orders and stock. TPC-H is an ad hoc decision support benchmark, simulating an application performing complex

[1] http://www.osdl.org/

[2] http://www.tpc.org/

business analyses that supports the making of sound business decisions for a wholesale supplier.

It is impractical for most Linux kernel developers to adhere to TPC rules, simply for cost alone. The executive summaries of the published results on the TPC Results Listing Web page [3] show system costs in the millions of dollars.

To illustrate, the highest performing TPC-W result at the 10,000 item scale factor[4] [TPCW10K] uses forty-eight dual-processor Web servers, twelve dual-processor and two single-processor Web caches, and one system with eight processors and approximately 150 hard drives for the database management system. This does not include the systems emulating Web browsers that drive the benchmark.

The highest performing TPC-C [TPCCRESULT] result uses thirty-two eight-processor system with 108 hard drives each, four four-processor systems with fourteen hard drives each for the database management system, and sixty-four dual-processor clients. This does not include the systems emulating the terminals required to drive the benchmark.

The highest performing TPC-H result at the 10,000 GB scale factor [TPCH10000GB] uses sixty-four dual-processor systems with four gigabytes of memory each and a total of 896 hard drives.

[3]Current results can be viewed on the Web at: http://www.tpc.org/information/results.asp

[4]The *scale factor* determines the initial size of the database.

## 2 Database Test Suite

The Database Test Suite[5] is a collection of simplified derivatives of the TPC benchmarks that simulate real-world workloads in smaller scale environments than that of a full blown TPC benchmark. These workloads are not very well suited to compare databases or systems because the variations allowed in running the workload would result in an apples-to-oranges comparison. However, these workloads can be used to compare the performance between different Linux kernels on the same system.

The amount of database administration knowledge and resources needed to run one of these workloads may still be intimidating to some, but the OSDL Scalable Test Platform[6] (STP) offsets these concerns. How the STP can be used is discussed towards the end of this paper.

These tests were initially developed on Linux with SAP DB but each test kit is designed to allow them to be usable with any database, such as MySQL[7] or PostgreSQL[8], with some porting work. Members of the PostgreSQL community are currently contributing to the Database Test Suite so that it may be used with PostgreSQL and also run on FreeBSD[9].

Each test provides scripts to collect system statistics using sar, iostat, and vmstat. Database statistics for SAP DB are also collected by using the tools that are provided with the database. The data presented for each workload in this paper are primarily profile data to show what parts of the Linux kernel is exercised. Keep in mind that the profile data

[5]http://www.osdl.org/projects/performance/

[6]STP can be access through the Web at http://www.osdl.org/stp/

[7]http://www.mysql.com/

[8]http://www.postgresql.org/

[9]http://www.freebsd.org/

presented here characterizes the workload for a specific set of parameters and system configurations. For example, as we review the profile data we will see that Database Test 2 appears to be stressing a SCSI disk controller driver, yet the workload can be customized so that the working set of data can fit completely into memory to put the focus of the workload on the system memory and processors as opposed to the storage subsystem.

### 2.1 Database Test 1 (DBT-1)

Database Test 1 (DBT-1) is derived from the TPC-W Specification [TPCW], which typically consists of an array of Web servers, hosting the on-line reseller's store front, that interfaces with a database management system, shown in Figure 1. An array of systems is also required to support the benchmark by simulating Web browsers accessing the Web site.

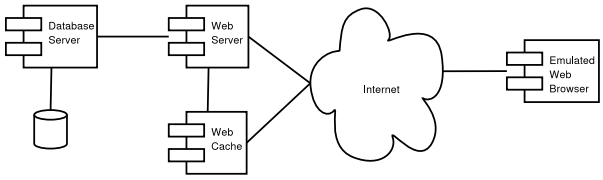


Figure 1: TPC-W Component Diagram

DBT-1, on the other hand, focuses on the activities of the database. There are no Web servers used, and a simple database caching application can be used to simulate some of the effects that a Web cache would have on the database. Since no Web servers are used, emulated Web browsers are not fully implemented. Instead, a driver is implemented to simulate users requesting the same interactions against the database that a Web browser would make against a Web server.

Figure 2 is a component diagram of the programs used in DBT-1. The Driver is a multi-threaded program that simulates users accessing a store front. The Application Server is another multi-threaded application that manages a pool of database connections and handles interaction requests from the Driver. The Database Cache is yet another multi-threaded program that extracts data from the database before the test starts running. If the caching component is not used, the Application Server queries the database directly. Each component of DBT-1 can run on separate or shared systems, in other words, in a three-tier or one-tier environment.

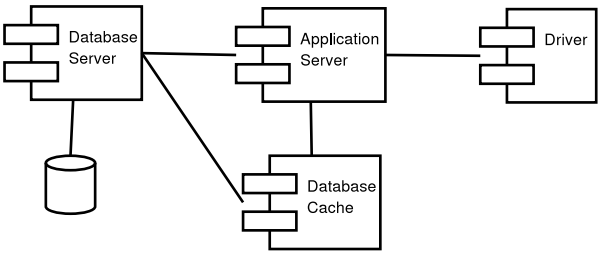


Figure 2: DBT-1 Component Diagram

The emulated users behave similarly to the TPC-W emulated browsers by executing interactions that search for products, make orders, display orders, and perform administrative tasks. There are a total of fourteen interactions that can occur, as shown in Table 1 with their frequency of execution. Each interaction, except Customer Registration, causes read-only or read-write I/O activity to occur. The overall effect is that 80% of the interactions executed cause read-only I/O activity while the remaining 20% of the interactions execute also cause writing to occur. Admin Request, Best Sellers, Home, New Products, Order Inquiry, Order Display, Product Detail, Search Request, and Search Results are the read-only interactions. Admin Confirm, Buy Request, Buy Confirm, and Shopping Cart are the read-write interactions. Customer registration does not interact with the database. It is maintained in the workload to keep the inter-

action mix close to the TPC-W specification.

| *Interaction* | *Executed* |
|---|---|
| Admin Confirm | 0.09 % |
| Admin Request | 0.10 % |
| Best Sellers | 11.00 % |
| Buy Confirm | 0.69 % |
| Buy Request | 0.75 % |
| Customer Registration | 0.82 % |
| Home | 29.00 % |
| New Products | 11.00 % |
| Order Display | 0.25 % |
| Order Inquiry | 0.30 % |
| Product Detail | 21.00 % |
| Search Request | 12.00 % |
| Search Results | 11.00 % |
| Shopping Cart | 2.00 % |

Table 1: DBT-1 Database Interactions Mix

An emulated user maintains state between interactions to simulate a browser session. A session lasts an average of fifteen minutes, which can be tester defined, over a negative exponential distribution. The state maintained includes the emulated user's identification and a shopping cart identifier. Each emulated user also randomly picks a think time from a negative exponential distribution, with a specified average, to determine how long to sleep between interactions.

This workload generally exhibits high processor and memory activity mixed with with networking and low to medium I/O activity that is mostly read-only. The parameters that can be controlled to alter the characteristics of the workload are the scale factor of the database, the number of connections the Application Server opens to the database, the number of emulated users created by the driver, and the think time between interaction requests.

The following results for DBT-1[10] are collected from a four-processor Pentium III Xeon* system with 1 MB of L2 cache and 4 GB of memory in the OSDL STP environment. The system is configured in a one-tier environment with SAP DB using a total of eleven raw disk devices and to run against Linux 2.5.67. A database with 10,000 items and 600 users is created, where 400 users are emulated using an average think think of 1.6 seconds

Table 2 displays the top 20 functions called sorted by the frequency of clock ticks. The profile data is collected using readprofile where the processors in the system become 100% utilized, as shown in Figure 3. Other than the scheduler, we can see that TCP network functions are called most frequently in this workload.

Table 3 displays the top 20 functions with the highest normalized load, which is calculated by dividing the number of ticks a function has recorded by the length of the address space the function occupies in memory. By comparing Table 2 and Table 3, we can see that `__copy_to_user_ll` and `__copy_from_user_ll` appear in both tables. This implies that the user address space is accessed frequently in this workload.

### 2.2 Database Test 2 (DBT-2)

Database Test 2 (DBT-2) is derived from the TPC-C Specification [TPCC], which typically consists of a database server and a transaction manager used to access the database server. There is also an array of systems required to support the benchmark by simulating terminals accessing the database. This benchmark can be run in one of two configurations, as shown in Figure 4 and Figure 5. The only difference be-

[10] http://khack.osdl.org/stp/271067/

| *Function* | *Ticks* |
|---|---|
| default_idle | 944946 |
| schedule | 15835 |
| __wake_up | 7376 |
| tcp_v4_rcv | 6871 |
| __copy_to_user_ll | 6456 |
| tcp_sendmsg | 6386 |
| mod_timer | 4666 |
| __copy_from_user_ll | 4482 |
| tcp_recvmsg | 4297 |
| __copy_user_intel | 3602 |
| tcp_transmit_skb | 3369 |
| ip_queue_xmit | 3230 |
| dev_queue_xmit | 3105 |
| ip_output | 2936 |
| __copy_user_zeroing_intel | 2806 |
| tcp_data_wait | 2711 |
| tcp_rcv_established | 2675 |
| generic_file_aio_write_nolock | 2614 |
| fget | 2536 |
| do_gettimeofday | 2420 |
| ... | ... |
| total | 1124848 |

Table 2: DBT-1 Profile Ticks

| *Function* | *Load* |
|---|---|
| default_idle | 14764.7812 |
| __wake_up | 153.6667 |
| __copy_to_user_ll | 57.6429 |
| system_call | 52.5682 |
| get_offset_tsc | 45.9688 |
| syscall_exit | 40.1818 |
| __copy_from_user_ll | 40.0179 |
| fget | 31.7000 |
| restore_fpu | 30.6875 |
| fput | 26.4062 |
| ipc_lock | 24.6250 |
| __copy_user_intel | 22.5125 |
| sock_wfree | 19.6094 |
| __copy_user_zeroing_intel | 17.5375 |
| local_bh_enable | 16.7396 |
| mod_timer | 16.2014 |
| schedule | 15.4639 |
| do_gettimeofday | 15.1250 |
| sockfd_lookup | 14.8393 |
| device_not_available | 13.4146 |

Table 3: DBT-1 Normalized Profile Load

tween these two configuration is that the emulated terminals access the database through a transaction manager, labeled as the *Client* in Figured 4, while the emulated terminals access the database directly in the Figure 5.

DBT-2 can be also be configured to run in one of two ways. The first way is shown in Figure 6 where the Driver, a multi-threaded program that creates a thread for every terminal emulated, accesses the database through the Client program, a transaction manager that is a multi-threaded program that manages a pool of database connections. The second way, shown in Figure 7, combines the functionality of the Client program into the Driver program so that the driver can connect directly to the database. In either case, the workload can be run in a single- or multi-tier environment.

DBT-2 consists of five transactions that create orders, display order information, pay for orders, deliver orders, and examine stock levels. Table 4 lists each transaction and the frequency that each is executed by the emulated terminals. The Deliver, New-Order, and Payment transactions are read-write transactions, while the Order-Status and Stock-Level transactions are read-only transactions.

This workload can be customized so that the working set of data is contained completely in memory. If the working set of data is cached completely in memory, the system puts a heavy load on processors and memory usage. In situations where the working set of data is not completely cached, random I/O activity increases, while in both cases, sequential writes

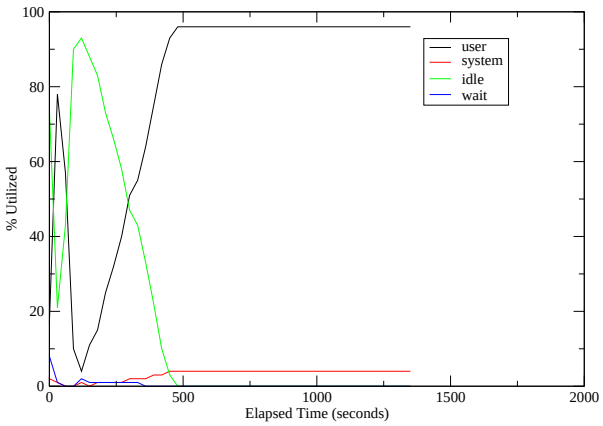


Figure 3: DBT-1 Processor Utilization

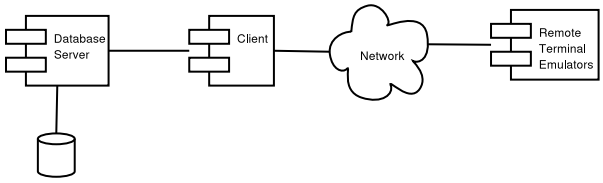


Figure 4: TPC-C Component Diagram 1

to the database logging device occurs throughout the test.

There are several parameters that can be customized to alter the characteristics of the workload. There are constant keying and thinking times between interactions that can be tester defined. The keying time simulates the time taken to enter information into a terminal and the thinking time simulates the time taken for a tester to determine the next transaction to execute.

By default, every terminal that is emulated is assigned a district and a warehouse to work out of. This can be changed so that a terminal randomly picks a warehouse and district in a specified range for every transaction. The effect this has on the workload is that a single emulated terminal is likely to access a greater amount of data in the database over the course of a test.

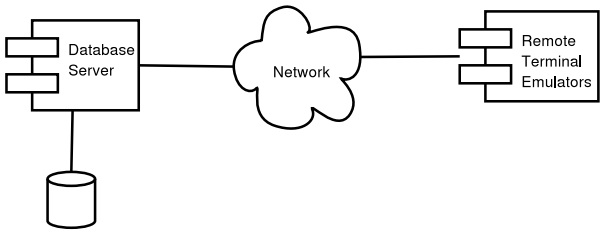


Figure 5: TPC-C Component Diagram 2

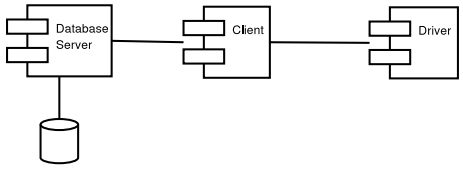


Figure 6: DBT-2 Component Diagram 1

Given the same amount of memory, this would create a workload less likely to be cached in memory and more likely to incur an increased amount of I/O activity. It would also create more lock contention in the database. By reducing the number of emulated terminals and by limiting the range of data an emulated terminal accesses in the database, the workload can be cached into memory, effectively creating a workload that only performs synchronous writes on the database logging device.

The following results for DBT-2[11] are collected from a four-processor Pentium III Xeon system with 1 MB of L2 cache and 4 GB of memory in the OSDL STP environment. The system is configured in a one-tier environment with SAP DB using twelve raw disk devices to run against Linux 2.5.67. Sixteen terminals are emulated to randomly select a district across six distinct warehouses with a keying and thinking time of zero seconds.

Table 5 displays the top 20 functions called

[11] http://khack.osdl.org/stp/271071/

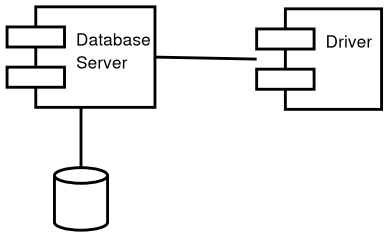


Figure 7: DBT-2 Component Diagram 2

| *Transaction* | *Executed* |
|---|---|
| Delivery | 4.0 % |
| New-Order | 45.0 % |
| Order-Status | 4.0 % |
| Payment | 43.0 % |
| Stock-Level | 4.0 % |

Table 4: DBT-2 Database Transaction Mix

sorted by the frequency of clock ticks and Table 6 displays the top 20 functions with the highest normalized load. If we compare these tables like we did with the results from DBT-1, we see that `bounce_copy_vec`, `__blk_queue_bounce`, `scsi_request_fn`, `scsi_end_request`, and `page_address` appear in both tables. The `default_idle` function also appears at the top of both tables. This implies that the processors on the system are not fully utilized and that the system may be stressing the storage subsystem. Figure 8 confirms that the processors are approximately 5% to 10% idle and are approximately 40% to 50% busy waiting for I/O.

### 2.3 Database Test 3 (DBT-3)

Database Test 3 (DBT-3) is derived from the TPC-H Specification [TPCH], which typically consists of a single database server that is

| *Function* | *Ticks* |
|---|---|
| default_idle | 5695427 |
| bounce_copy_vec | 84136 |
| schedule | 55663 |
| __blk_queue_bounce | 28391 |
| scsi_request_fn | 23052 |
| do_softirq | 21760 |
| __make_request | 20124 |
| try_to_wake_up | 10511 |
| scsi_end_request | 10161 |
| system_call | 9734 |
| dio_bio_end_io | 9241 |
| scsi_queue_next_request | 9066 |
| ipc_lock | 6856 |
| sys_semtimedop | 5858 |
| do_anonymous_page | 5693 |
| kmem_cache_free | 5587 |
| free_hot_cold_page | 4768 |
| page_address | 4752 |
| buffered_rmqueue | 4648 |
| try_atomic_semop | 4406 |
| ... | ... |
| total | 6211231 |

Table 5: DBT-2 Profile Ticks

queried by an application in a host-based or client/server configuration, as shown in Figure 9 and Figure 10.

There are twenty-two queries that provide business analyses for pricing and promotions, supply and demand management, profit and revenue management, customer satisfaction, market share, and shipping management. In addition to the twenty-two queries, there are two refresh functions that load new sales information into the database.

This workload consists of loading a database, running a series of queries against the database, and loading new sales information into the database. There are three distinct tests in which these actions occur, the Load Test, Power Test,

| *Function* | *Load* |
|---|---|
| default_idle | 88991.0469 |
| bounce_copy_vec | 1051.7000 |
| system_call | 221.2273 |
| syscall_exit | 105.5455 |
| do_softirq | 104.6154 |
| ipc_lock | 85.7000 |
| dio_bio_end_io | 82.5089 |
| kmem_cache_free | 69.8375 |
| scsi_end_request | 63.5063 |
| __wake_up | 55.9375 |
| schedule | 54.3584 |
| get_offset_tsc | 54.1562 |
| __blk_queue_bounce | 47.9578 |
| bio_put | 46.3542 |
| scsi_request_fn | 42.3750 |
| fget | 37.5125 |
| restore_fpu | 36.7812 |
| page_address | 33.0000 |
| generic_unplug_device | 29.7143 |
| device_not_available | 29.6098 |

Table 6: DBT-2 Normalized Profile Load

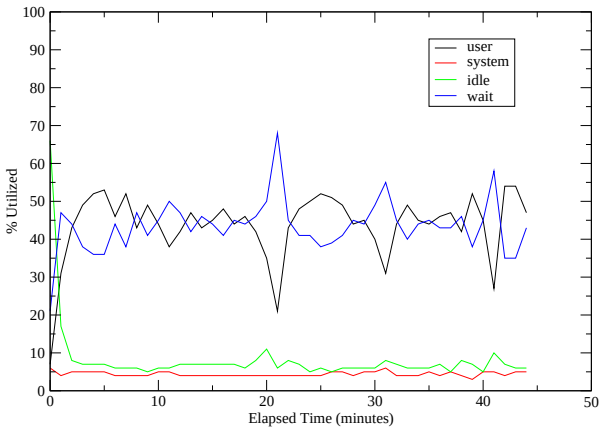


Figure 8: DBT-2 Processor Utilization

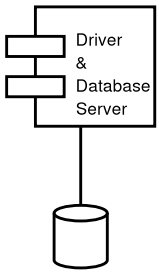


Figure 9: TPC-H Host-Based Component Diagram

and Throughput Test. The Load Test creates the database tables and loads data into them. The Power Test executes each of the twenty-two queries and two refresh functions sequentially. The Throughput Test executes a specified number of processes that executes each of the twenty-two queries in parallel and an equal number of processes that executes only the refresh functions.

There are several ways that this workload can be customized. The scale factor of the database can be selected so that at some point during a test, the working set of data becomes cached into memory. The number of streams for the throughput test may have to be adjusted according to the available system resources. The TPC-H Specification requires a minimum number of streams to be used depending on the scale factor of the database and ideally the number of streams should be selected so that the highest throughput metric can be achieved. However, for DBT-3, selecting the number of streams can be determined by how the Linux kernel is stressed by the workload. In any case, if the working set of data is not cached, large sequential I/O activity occurs in the Power and Throughput Test. Also, each of the twenty-two queries can be modified to meet different needs. For example, a query can be modified to answer another type of business question.

The following results for DBT-3[12] are collected from a four-processor Xeon* system with 256 KB of L2 cache and 4 GB of memory in the OSDL STP environment. The system is con-

[12] http://khack.osdl.org/stp/271071/

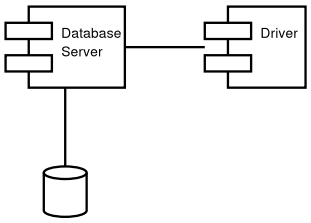


Figure 10: TPC-H Client/Server Component Diagram

figured in a host-based environment running Linux 2.5.67 with hyper-threading enabled and a patch that allows the DAC960 driver to have direct memory access into high memory. A 1 GB database was created and only statistics from a Throughput Test with eight streams are reported here.

Table 7 displays the top 20 functions called sorted by the frequency of clock ticks. Similar to DBT-2, the prominence of `default_idle` with `__make_request` and `DAC960_BA_InterruptHandler` suggests that the system is also busy waiting for I/O. Table 8 displays the top 20 functions with the highest normalized load and again, similar to DBT-2, seeing `DAC960_BA_InterruptHandler` as one of the more prominent functions on this list also supports the theory that the kernel is spending a significant amount of time attempting to process I/O requests. Figure 11 shows that the processors are waiting for I/O 40% to 80% of time throughout the middle of the Throughput Test.

| *Function* | *Ticks* |
|---|---|
| poll_idle | 24160697 |
| __make_request | 21161 |
| schedule | 16354 |
| generic_unplug_device | 14701 |
| DAC960_LP_InterruptHandler | 12160 |
| system_call | 7361 |
| kmap_atomic | 3783 |
| fget | 3148 |
| get_user_pages | 3099 |
| do_direct_IO | 2968 |
| dio_await_one | 2955 |
| bio_alloc | 2850 |
| device_not_available | 2767 |
| direct_io_worker | 2551 |
| blockdev_direct_IO | 2226 |
| find_vma | 2020 |
| __generic_file_aio_read | 1924 |
| follow_page | 1891 |
| __copy_to_user_ll | 1817 |
| ... | ... |
| total | 24322403 |

Table 7: DBT-3 Profile Ticks

## 3 PLM and STP

Using the Database Test Suite can be an intimating task for those inexperienced with administering database management systems, or large systems may not be readily available for testing. Rather than implementing one of the Database Test Suite workloads on their own system, Linux kernel developers can test their kernel patches by using the Patch Lifecycle Manager[13] (PLM) and the STP. In order to use PLM or STP, you must sign up as an associate of the OSDL, free of charge, through the Web at `http://www.osdl.org/`.

PLM can be used to store patches for the Linux kernel that can be used by STP for testing. Currently, PLM automatically copies Linus Torvalds's tree as well as Andrew Morton's, Martin Bligh's, Alan Cox's, and the ia64 patch sets. PLM also executes filters against each patch entered into the system, to verify the patch ap-

[13]PLM can be accessed through the Web at http://www.osdl.org/cgi-bin/plm/

| *Function* | *Load* |
|---|---|
| poll_idle | 383503.127 |
| system_call | 167.2955 |
| generic_unplug_device | 140.0095 |
| DAC960_LP_InterruptHandler | 74.1463 |
| device_not_available | 67.4878 |
| fget | 44.9714 |
| kmap_atomic | 34.3909 |
| fput | 31.1154 |
| find_vma | 24.3373 |
| unlock_page | 20.9059 |
| restore_fpu | 20.4857 |
| get_offset_tsc | 19.6667 |
| syscall_call | 19.4545 |
| io_schedule | 18.8542 |
| __make_request | 18.7431 |
| dio_await_one | 18.5849 |
| current_kernel_time | 18.5303 |
| math_state_restore | 17.7846 |
| mempool_alloc_slab | 15.9524 |
| kmem_cache_alloc | 15.8158 |

Table 8: DBT-3 Normalized Profile Load

plies to a kernel or another patch, and verifies that the kernel can still be compiled with that patch.

STP currently implements all three of the workloads in the Database Test Suite[14]. DBT-1 can be run on systems with 2, 4 or 8 processors, DBT-2 and DBT-3 can be run on systems with 4 processor. The 4 and 8 processor systems also have arrays of external hard drives attached. Each test in STP generates a Web page of results with links to raw data and charts, as well as profile data if desired. E-mail notification is also sent to the test requester when a test has completed.

[14]DBT-3 is currently being developed for STP and should be available by the time this paper is published.

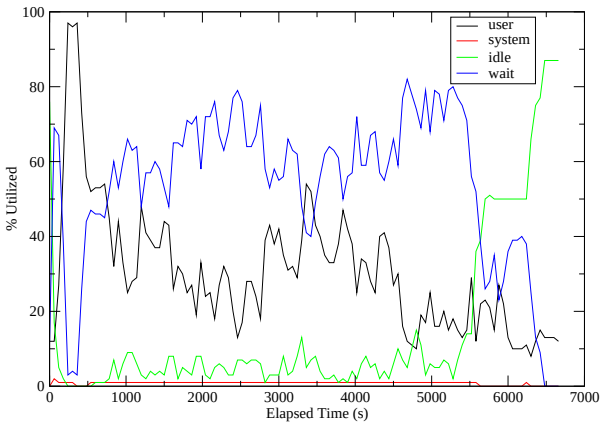


Figure 11: DBT-3 Throughput Test Processor Utilization

## 4 Comparing Results

While the OSDL Database Test Suite is derived from TPC benchmarks, results from the Database Test Suite are in no way comparable to results published from TPC benchmarks. Such comparisons should be reported to the TPC (admin@tpc.org) and to the OSDL (wookie@osdl.org).

## References

[TPCC] *TPC Benchmark C Standard Specification Revision 5.0*, February 26, 2001.

[TPCH] *TPC Benchmark H Standard Specification Revision 1.5.0*, 2002.

[TPCH10000GB] *NCR 5350 Using Teradata V2R5.0 Executive Summary*, Teradata a division of NCR, March 12, 2003.

[TPCCRESULT] *ProLiant DL760-900-256P Client/Server Executive Summary*, Compaq Computer Corporation, September 19, 2001.

[TPCW] *TPC Benchmark W Specification Version 1.6*, August 14, 2001.

[TPCW10K] *Netfinity 5600 with Netfinity 6000R using Microsoft SQL Server 2000 Executive Summary*, International Business Machines, Inc., July 1, 2000.

## Trademarks