



Infrastructure Technology
Product of the Year

XenRT

XenSource's Xen testing infrastructure

James Bulpin, XenSource Inc.
james@xensource.com

Xen Summit 8/Sep/2006

www.xensource.com



What is XenRT?

- An infrastructure for testing Xen
 - Functional, regression and performance
- Used for testing:
 - Source distributions
 - xen-unstable
 - xen-3.0-testing
 - RPMs, including vendor kernel ports
 - Official vendor ports
 - Unofficial XenSource ports (<http://xenbits.xensource.com/kernels>)
 - Binary distributions
 - Point release snapshots
 - XenEnterprise
 - Private branches/patches



XenRT uses

- Regular regression testing
- “patchman” sanity tests before pushing code
- Test patches before committing to trees
- Performance testing and investigation
- Hardware compatibility testing
- Preparing test machine for manual testing



Infrastructure (1)

A XenRT deployment has:

- A job and result database and HTTP interface
- A controller host to select and execute jobs
- Standard DNS, DHCP and TFTP servers
- One or more build servers
- Serial console servers
- Facilities for power cycling test machines
- A number of test machines



Infrastructure (2)

XenRT needs access to:

- Repositories of
 - Linux distribution images as tarballs
 - Linux distribution RPMs
 - Windows unattended installation ISOs
 - Canned Windows disk images
- Benchmark packages



Terminology

Test: a benchmark or functional test entity that contains one or more test cases

- e.g. LTP, Imbench, xm-test, SPEC JBB

Phase: a grouping of one or more tests for convenience of reference

- e.g. Phase 1 could be Domain-0 tests, Phase 2 could be non-SMP Linux guest tests

Sequence: one or more phases executed in serial, parallel or a combination of both

- e.g. The “full” sequence runs phases for Domain-0 tests, non-SMP guests, SMP guests, etc..

Job: a unit of testing work that runs a sequence with a particular configuration on a machine



Jobs

- Jobs are managed by a central or per-site job scheduling database.
- Users or automated processes submit jobs with parameters
 - what to test, e.g. tree and changeset
 - system configuration, e.g. SMP dom0, guest distribution, guest configs etc.
 - constraints for selecting machine, e.g. “memory>=4G”
- Per-site job manager daemon searches for suitable jobs for idle test machines
- User can poll, monitor and receive notification of completion and browse/extra results



Sequences

- Specify tests and phases
 - Each test instance has:
 - Test to run
 - Where to run (Domain-0, which guest, etc.)
 - Optional arguments to test script
- Dependencies between tests
 - Test start can depend on other test(s) having started, finished and/or passed
 - Build arbitrary test graphs
- Define over-all sequence pass criteria



Current standard tests

bonnie++	burnintest	crashme	dbench
iometer	iozone	kernbench	lmbench
ltp	memtest	osdb	osldaim
postmark	prime95	sandra	sciencemark2
sio	sysmark04	specjbb	sqlbench
tbench	ttcp	wine	xm-test
dvdstore	httperf	HCT	loadsim
...			



Current in-house tests

srm: heavy duty save/restore/migrate

vm86: unit test for this mode

timecheck: sanity check for dom0 time

installvm: prepare a guest image

buildxen: build from source

anytest: parallel random stress testing

churntest: rapid and parallel VM create/shutdown

netcheck: guest, dom0 and off-box connectivity

...



Preparing a machine

- Assume disk is in an undefined state
- PXE boot with ramdisk root filesystem
- fdisk if necessary
- mkfs Domain-0 partition(s)
- Untar Domain-0 filesystem
- Tailor filesystem as necessary
- Install Xen, either:
 - ./install.sh from dist directory
 - If prebuilt RPMs: rpm --install
- XenEnterprise unattended install



Creating guest images

- “Legacy” Linux guests
 - Create LVM/loopback/NFS volume
 - mkfs
 - Untar chosen distribution
 - Tailor image (network configs etc.)
- New Linux guests
 - Create fresh installation from RPM repository
- Windows guests
 - Create an empty backing volume
 - Run an unattended Windows install from ISO
 - Alternatively dd a canned image



Test Dispatch (1)

- Execute tests based on dependencies
 - Initially based on makefiles
 - Moving to XML-based config
- Dispatcher connects to execution location using SSH, invokes actions on test script:
 - **install**: install test into virtual machine
 - **start**: start test asynchronously
 - **waitfor**: monitor test for completion and liveness
 - **process**: extra outcome, results etc. in standard format
 - **getlogs**: retrieve log files for possible triage
 - **cleanup**: remove working/temp files etc.

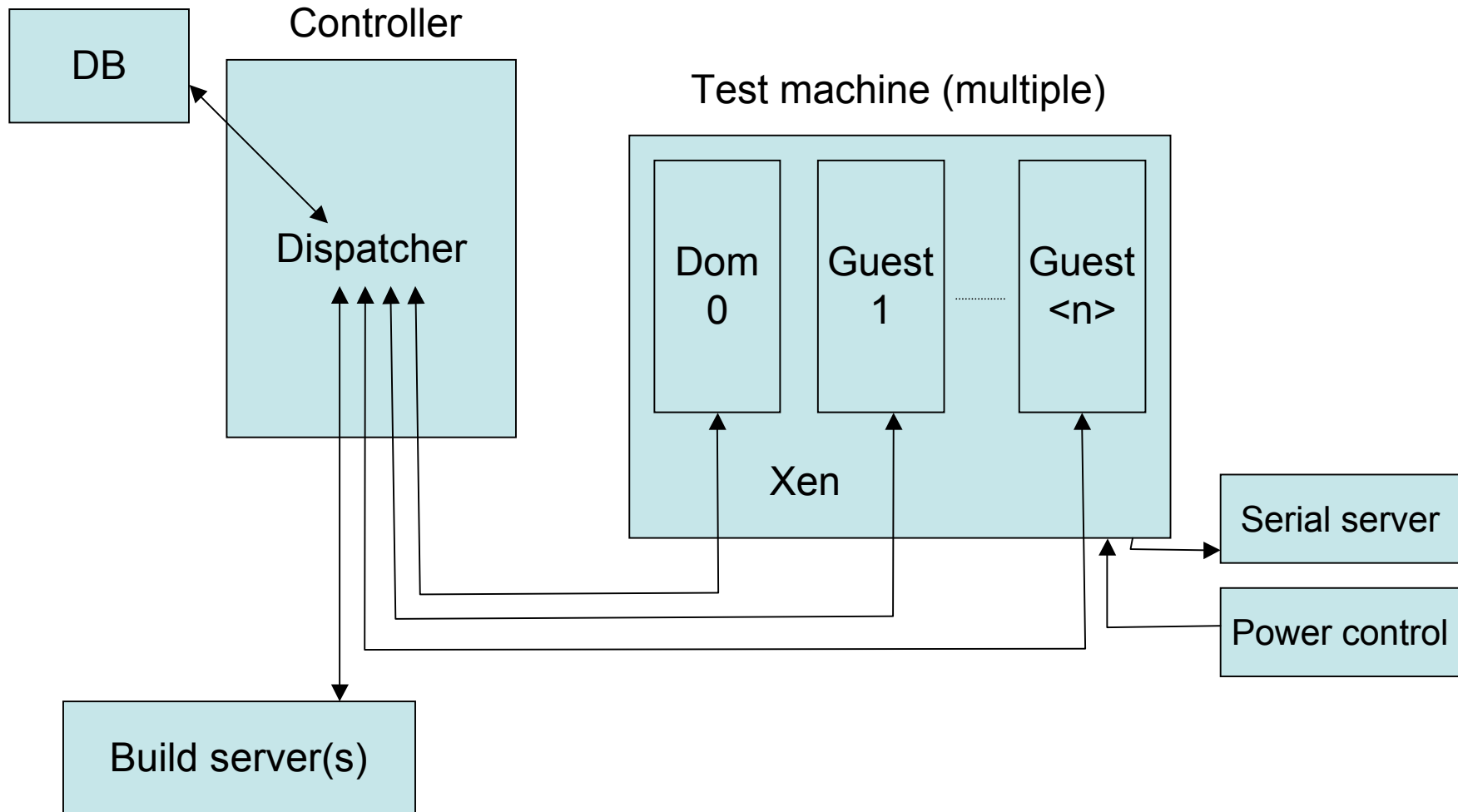


Test Dispatch (2)

- Dispatcher can coordinate tests across:
 - Multiple guests and/or Domain-0
 - Multiple machines, physical, virtual or native
 - E.g. for TTCP and httperf inter-machine tests
 - E.g. for migration
- Test monitoring
 - Liveness of tests, timeouts
 - Monitor the monitoring script
 - Running harness off-box make it easier to deal with failures



Execution architecture





Recording data

- Postgres database
 - All job parameters
 - Test data
 - Current status and outcome
 - Numeric results
 - Comments and failure descriptions
- Log files
 - Test-specific logs
 - syslog, dmesg for each test
 - Dispatcher logs
 - Serial console logs
 - Guest console logs



Test matrix (1)

- The Xen test matrix is huge
- Platform (hypervisor, dom0, tools, host):
 - x86-32, PAE, x86-64, (PPC, IA64)
 - Hardware: CPU, disk interface, NIC, < or > 4GB
 - VT or AMD-V
 - dom0 vCPU count
 - dom0 autoballooning vs. fixed allocation
 - dom0 Linux distribution
 - Xen or dom0 boot options



Test matrix (2)

- Each guest
 - x86-32, PAE, x86-64 (if allowed to differ from dom0)
 - HVM or paravirtualised
 - vCPU count, memory size
 - LVM, file-backed, blktap, NFS root, iSCSI, GNBD
 - Linux distribution, Windows version
 - If 1 vCPU, CONFIG_SMP kernel?
 - Number of VBDs, VIFs
 - For PV, same kernel as dom0 or different?
 - Guest kernel version same or older than dom0 (e.g. 3.0.1 on unstable)
- Entire test
 - Number of guests to have running in parallel
 - Combination of guests types to run
 - CPU pinning

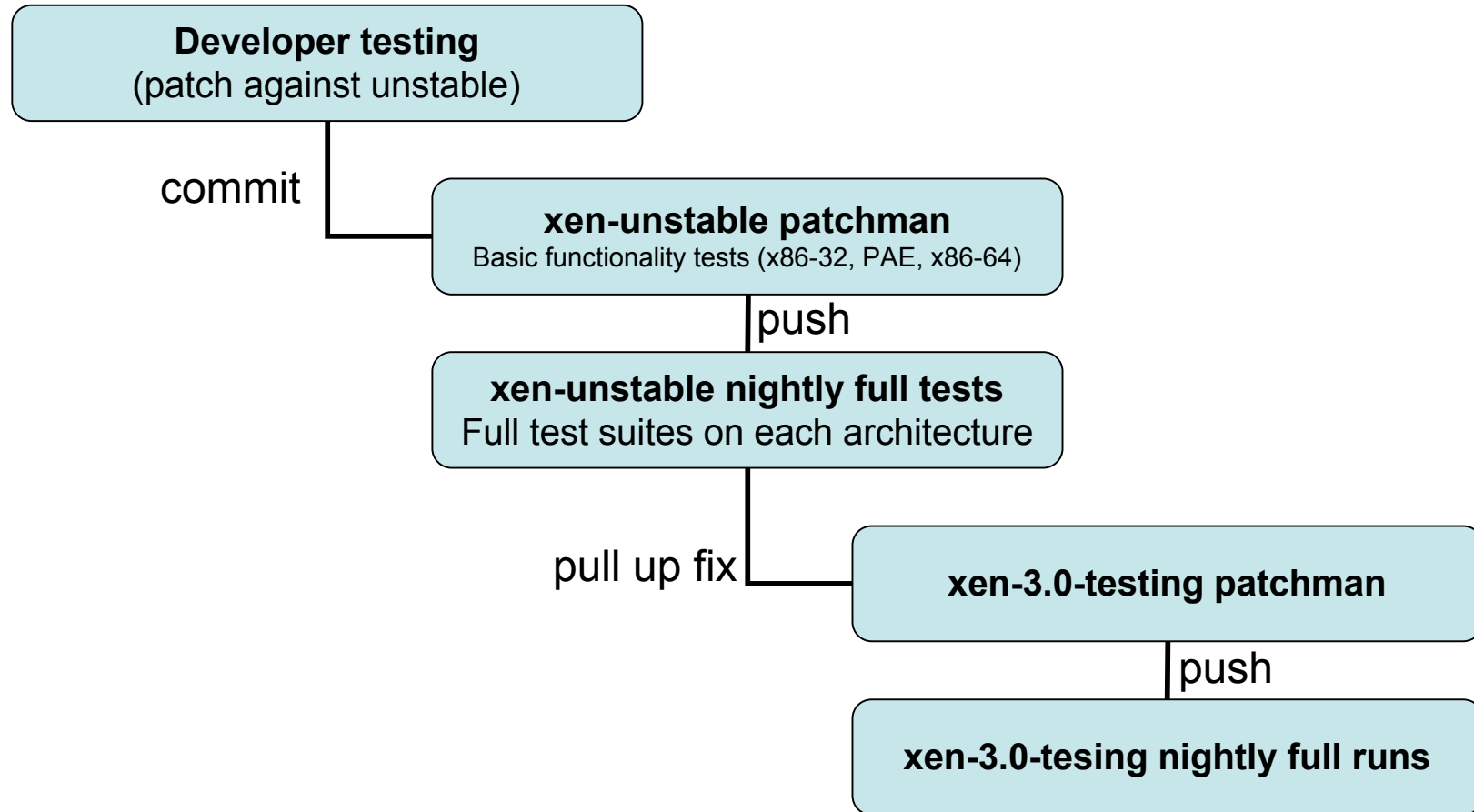


Interesting points in the matrix

- Most regular tests use LVM, vcpus=1 dom0
- Hardware config is allowed to vary randomly
- Paravirtualised guests on a vcpus=1 dom0
 - vcpus=1 and vcpus=<phys cpus> guests
 - Separate job for each of x86-32, PAE and 64
 - Guests tested one at a time then in parallel
 - Use same –xen kernel as dom0
- HVM guests
 - vcpus=1 and vcpus=<phys cpus> guests
 - Linux and Windows guests
 - Each combination tested on VT and AMD-V
 - Separate job for each of x86-32, PAE and 64
 - Usually like on like, e.g. PAE on PAE
- Backwards compatibility
 - Single sequence with one guest each of 3.0.0, 3.0.1 and 3.0.2 on unstable



Testing basic PV functionality





Guests tested

- Linux
 - RHEL 3.x, RHEL 4.x
 - SLES 9.x, SLES 10
 - Debian
 - Fedora Core 5
- Windows
 - Windows Server 2003 Standard/Enterprise
 - Basic, SP1 and R2
 - Windows 2000
 - Windows XP SP2



Hardware

- Central test pool in Palo Alto
 - Server class hardware
 - Covering several vendors, chipsets, disk configurations, memory sizes, NICs etc.
 - Some given/loaned by vendors
 - Mostly purchased
 - Pre-release hardware (VT, AMD-V)
 - Currently 43 machines
 - Job scheduler maximises utilisation
- Small test pool in Cambridge



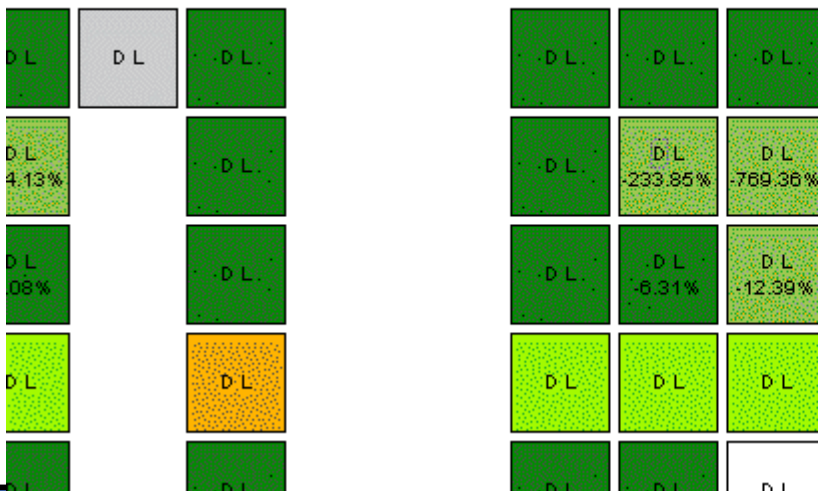
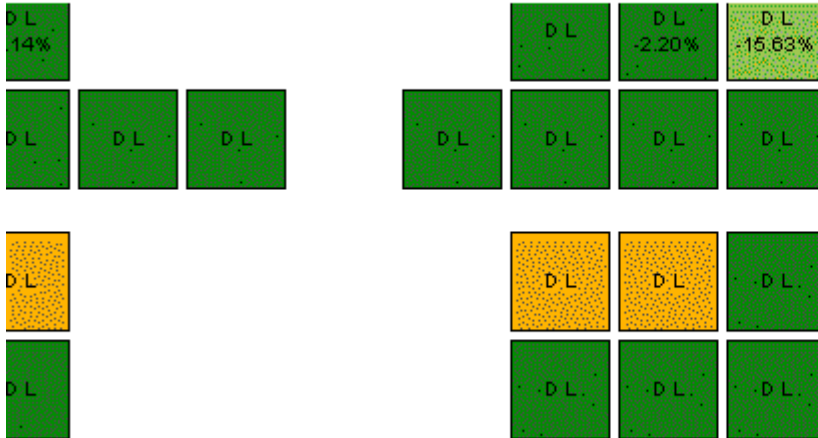
Data Browser

- Matrix of jobs X phases/tests
- Element summarises test outcome
 - Drill down for results, history
 - Per-test log browser
- Filter jobs by machine, architecture, job owner, Xen version, revision
- Application specific views for HCL, unstable tests, patchman
- CLI for scriptable data extraction
- Store native results for comparison
 - Visual warning if significant performance loss



Web interface

autorun xen-unstable CP Rev X



```

CHECK      ERROR
UPLOADED   yes
CHANGESET  1de184deaaa9c
RETURN     OK
OPTION_REMOVE_PASSED yes
STARTED    Wed Sep 6 20:30:04 PDT 2006
OPTION_DEBUG yes
USERID     autorun
JOBSTATUS  done
JOBID      12593
MACHINE    ultiar
FINISHED   Thu Sep 7 09:52:35 PDT 2006
VERSION    xen-unstable
RUNDIR     /usr/groups/xen/xenrt/production/share/re
RESOURCES_REQUIRED disks=1
OPTIONS    x86-32p

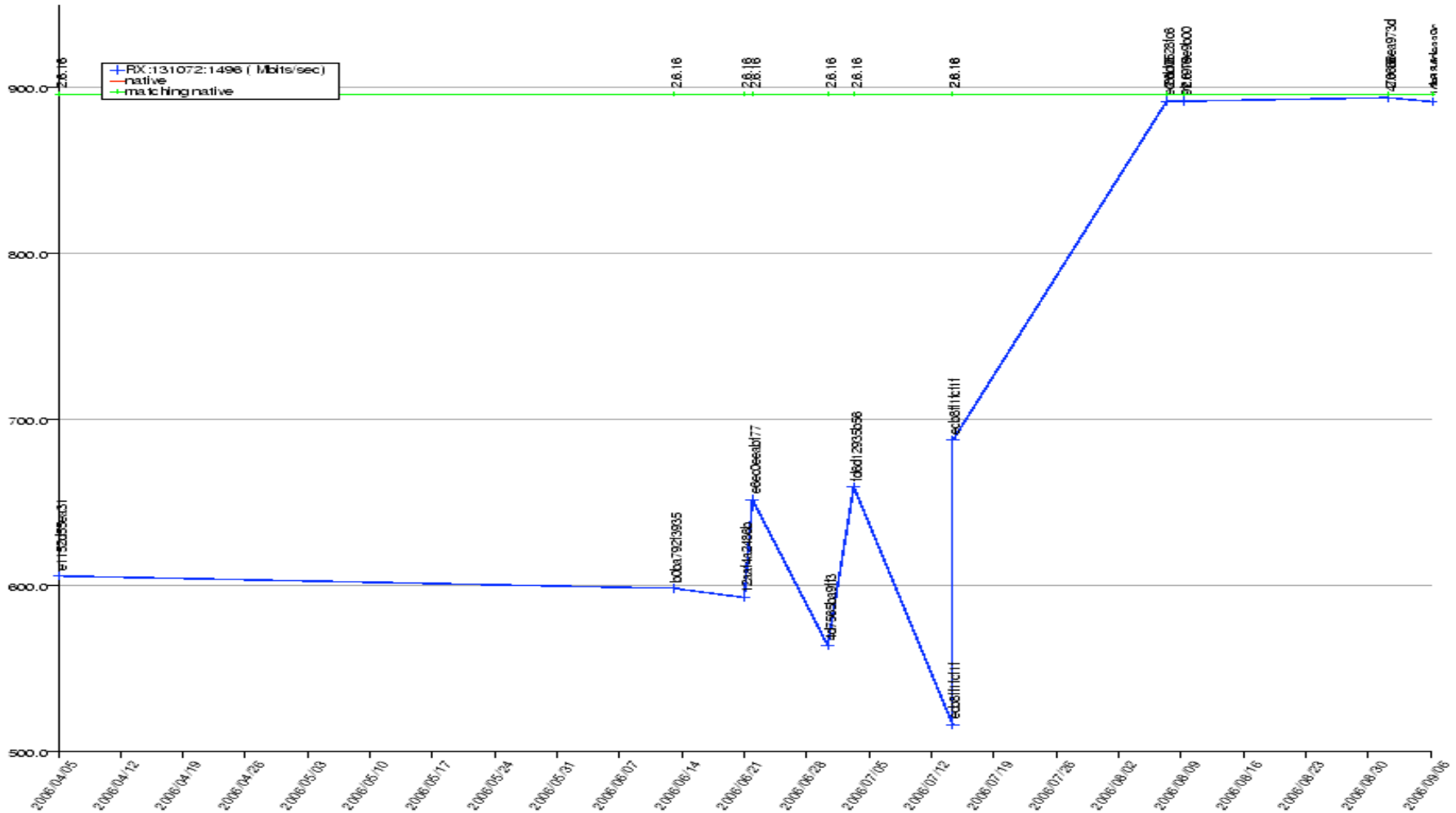
```

Phase 2 Imbench pass

Date / Time	Key	Value
2006-09-07 06:22:16	result	installing
2006-09-07 06:22:40	result	started
2006-09-07 06:22:40	result	installed
2006-09-07 07:06:03	result	pass
2006-09-07 07:06:03	V:LM_PROTFLT	1.12640 microsecond (-49.79%)
2006-09-07 07:06:03	V:LM_SYSCALL	0.34610 microsecond (+11.48%)
2006-09-07 07:06:04	V:LM_FORK_EXEC	1272.00000 microsecond (-172.04%)
2006-09-07 07:06:04	V:LM_FORK_EXIT	418.00000 microsecond (-233.85%)
2006-09-07 07:06:04	V:LM_PGFLT	6.36390 (+263.38%)
2006-09-07 07:06:05	V:LM_OSREAD	0.50210 microsecond (+7.72%)



Performance history





Stats (since March 2006)

- Jobs executed: 12,000
- Tests executed: 130,000
- Hours of testing (est.): 45,000
- Numerical results recorded: 280,000
- Logs captured: 18GB (compressed)



Ongoing work

- Adding new benchmark wrapper scripts
- New bespoke test cases
- Improving suite installation process
- New dispatcher, XML config
- Improving auto-triage
- Data browsing enhancements
- Documentation



Improving

- Analysis of data, particularly performance
- Broadened hardware coverage
 - Loan kit from vendors always welcome





Open source to-do list

- Sanitise scripts and config files for passwords
- Audit benchmark licenses for redistribution
- Lots and lots of documentation



Public data browser

- Had basic system in the run up to 3.0.0
 - No resource (technical and human) to maintain
- Effort has gone on the more feature-rich browser
 - Heavy duty database backend
 - Needs work on scaling
- To use this publicly:
 - Need to reduce/manage resource usage
 - Need to sanitise internal passwords from logs