Для цитирования: Рыбаков С. А. Автоматизация тестирования ОС «Эльбрус» на основе системы Autotest // Вопросы радиоэлектроники. 2017. № 3. С. 52–56.
УДК 004.415.538

С. А. Рыбаков[1]

[1] АО «МЦСТ»

АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ ОС «ЭЛЬБРУС» НА ОСНОВЕ СИСТЕМЫ AUTOTEST

В статье представлена система автоматизированного тестирования ядра ОС «Эльбрус», спроектированная на основе открытого проекта Autotest, реализованного и развиваемого рядом ведущих зарубежных корпораций. Новые системные решения и функциональности, введенные в Autotest при разработке системы, способствуют процессу постоянного усовершенствования версий операционной системы на базе ОС Linux, создаваемых специалистами АО «МЦСТ» для оригинальной и стандартных высокопроизводительных компьютерных платформ.

Ключевые слова: *система тестирования ОС Эльбрус, автоматизированное тестирование, Autotest.*

Введение

Система тестирования – это больше, чем набор простых функциональных тестов, это также и идентификация трендов (в т.ч. падения) производительности с добавлением статистического анализа [1]. Для качественного тестирования необходим широкий спектр тестов загрузки, регрессии, функциональности, производительности – от стрессовых тестов, создающих большую нагрузку на память или процессор, до тестов, интенсивно работающих с сетью.

Система тестирования операционной системы (СТОС) «Эльбрус», предназначенная для применения в процессе разработки, испытаний и эксплуатации ОС, включает в себя множество тестовых пакетов, созданных в АО «МЦСТ», а также универсальные утилиты, используемые при написании и запуске тестов. Она расширяется внешними системами тестирования, которые не опираются на особенности той или иной архитектуры, а потому могут быть использованы при тестировании ОС «Эльбрус» с минимальными изменениями. На данный момент в СТОС используются три внешние системы тестирования: Linux Test Project, Open POSIX Test Suite и UnixBench.

Linux Test Project (LTP) – это совместный проект компаний SGI, IBM, Cisco, Fujitsu, SUSE, Red Hat и других, предназначенный для проведения комплексной проверки компонентов и подсистем ОС на базе ядра Linux. LTP включает в себя тестирование: системных вызовов; подсистем управления памятью, планирования исполнения процессов, управления межпроцессорным взаимодействием, ввода-вывода, управления файловыми системами; сетевой подсистемы; неоднородного доступа к памяти (NUMA); математических операций и других.

Система Open POSIX Test Suite предназначена для аттестационного, функционального и нагрузочного тестирований на соответствие испытываемой системы спецификации системных интерфейсов POSIX (без учета особенностей конкретной реализации). Пакет тестов обеспечивает проверку очередей сообщений, семафоров, сигналов, потоков исполнения, таймеров и других объектов.

UnixBench – это пакет тестов, предназначенный для тестирования вычислительной системы и определения интегрального коэффициента, характеризующего ее быстродействие. В ходе тестирования измеряются производительность процессора, скорость выполнения операций с плавающей точкой и копирования файлов, скорость порождения процессов, характеристики межпроцессного взаимодействия, накладные расходы, связанные с переключением из пространства пользователя в пространство ядра, и некоторые другие показатели. В результате определяется суммарный коэффициент, который показывает, во сколько раз производительность тестируемой системы превосходит производительность системы, выбранной в качестве базовой.

Для ускорения поиска, анализа и исправления ошибок необходимо непрерывно запускать тесты на большом количестве машин, и единственный способ достичь этого заключается в полной автоматизации тестирования. Для ядра Linux проблема полного тестирования усложняется разнообразием поддерживаемого аппаратного обеспечения и большим количеством вариантов конфигурации ядра.

При принятой последовательности операций тестирования ОС «Эльбрус» оператору необходимо:

- установить на тестовую машину очередную версию компонентов ОС «Эльбрус», включающих в себя доработки общего программного обеспечения ОС (ОСОПО), в т.ч. ядра и модулей операционной системы;
- перезапустить машину с обновленным ядром ОС;
- запустить используемую систему тестирования ОС, проанализировать полученные результаты, скопировать их в архив и зарегистрировать соответствующие ошибки.

Данная схема усложняется наличием в каждой версии тестируемых компонентов ОС «Эльбрус» нескольких сборок ядра ОС. В зависимости от поддержки подсистемы неоднородного доступа к памяти (NUMA) и реального времени (RT) собираются четыре ядра ОС, каждое из которых следует протестировать. Более того, компоненты ОС собираются для каждой архитектуры в отдельности («Эльбрус», SPARC, Intel x86), а потому каждую сборку необходимо тестировать на машине соответствующей архитектуры. Поэтому для полного тестирования новой версии программного обеспечения оператору необходимо запускать систему тестирования ОС на разных машинах по нескольку раз (для всех конфигураций ядра ОС). Учитывая, что каждый запуск может занимать более часа, необходимость автоматизации тестирования становится очевидна.

В качестве основы для проведения такой работы был выбран открытый проект Autotest. Его определяющими целями стали реализация и внедрение системы автоматического тестирования ОС, позволяющей выполнить проверку конкретных версий ядра, разработанных для архитектур «Эльбрус», SPARC и Intel x86.

Принципиальные решения

При разработке ядра Linux каждая отдельная правка обычно не является большой, а установление авторства не составляет труда. Очевидно, чем раньше фиксируется незадача в правке, тем меньше от нее последствий и тем точнее будет отчет об ошибке и при ее исправлении. В идеале вклады отдельных разработчиков следует полностью проверить перед применением, но это непрактично, т.к. у большинства из них нет доступа ко всему спектру подлежащего тестированию оборудования. В связи с этим необходимо дать им доступ к большому набору машин с различным аппаратным обеспечением и скоординировать этот доступ. Для решения подобной проблемы и применяется Autotest – инструмент с открытым кодом, основанный на взаимодействии клиент-сервер, который создан и развивается Google, IBM, Red Hat и другими компаниями. Для его написания разработчиками был выбран Python – объектно-ориентированный язык с простым и понятным синтаксисом, мощной обработкой ошибок и исключений. Среди крупных проектов, использующих Autotest в качестве главной системы автоматизации тестирования, можно отметить AutoQA в Fedora, ОС Chrome, KVM и Goobuntu.

Autotest позволяет [2]:

- устанавливать и поддерживать контроль над клиентами по ssh-соединению;
- перезагружать клиентов и загружаться с новым ядром ОС;
- запускать тесты на клиентах и забирать результаты тестов обратно на сервер;
- обрабатывать большинство возникающих ошибок;
- при необходимости пытаться автоматически восстановить работоспособность клиентов;
- взаимодействовать с системой тестирования через два взаимозаменяемых интерфейса: графический Web Frontend и интерфейс командной строки.

Эти широкие возможности обусловили выбор Autotest в качестве базового средства автоматизации системы тестирования ОС «Эльбрус». Наряду с ними было решено добавить функциональность по обновлению ОСОПО, запуску СТОС и обработке результатов тестирования с последующим сохранением их в архиве.

Принципиальная схема созданной системы представлена на рис. 1.

Сервер Autotest устанавливается в виртуальном окружении, чтобы обеспечить возможность удаленного исправления неисправностей и обновления версий системы. Запуск задач системы возможен как локально – с любого компьютера из открытой подсети, так и удаленно – через VPN-подключение. Оба этих способа используют интерфейс Autotest Web Frontend. Задачи системы могут быть также поставлены автоматически с помощью интерфейса командной строки (например, планировщиком задач Cron). Для корректной работы на всех тестовых машинах должна быть установлена ОС «Эльбрус».

Тестовые машины могут входить в закрытую либо в открытую подсети. Машины закрытой подсети специально выделены для тестирования, на них не запускаются другие задачи. Кроме того, эти машины обладают следующими преимуществами:

- Удаленное управление питанием.
- Сбор диагностики с COM-портов.
- Тестирование производительности сетевого обмена.
- Контроль за составом аппаратуры, ПО и пользовательских настроек.

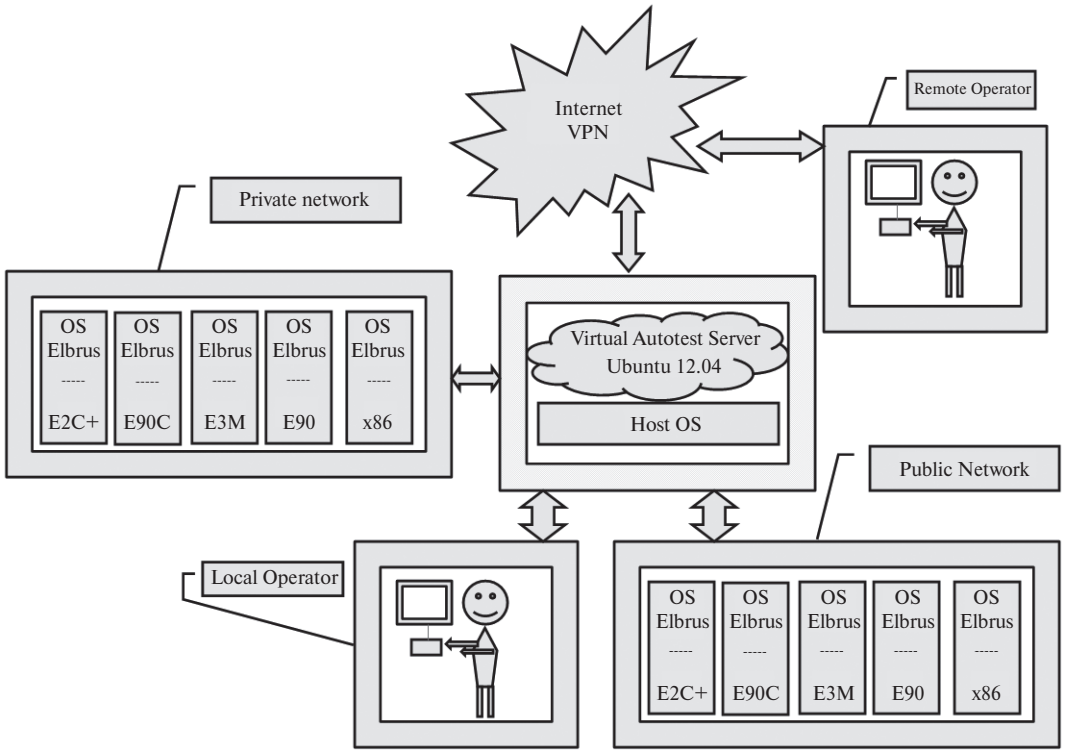

Рисунок 1. Схема системы автоматизированного тестирования ОС «Эльбрус» на основе Autotest

В закрытой подсети есть машины с различными архитектурами: «Эльбрус», SPARC, x86. Система автоматизированного тестирования использует по умолчанию именно эти машины.

Машины открытой подсети – это все остальные машины в составе сети, доступные серверу Autotest. Его доступ к этим машинам дает возможность запустить систему на любой машине с ОС «Эльбрус». Это может пригодиться, например, для тестирования новых аппаратных проектов.

Добавление машины (как закрытой, так и открытой подсети) в систему автоматизации тестирования не составляет труда: все, что необходимо сделать оператору, – установить беспарольное ssh-соединение между сервером Autotest и этой машиной (новым клиентом). Установка на клиента производится автоматически, если при обработке задачи сервером обнаруживается, что Autotest на клиенте не установлен.

Проблемы разработки

Самыми существенными проблемами описываемого проекта стали:

- обработка аварийного завершения системы тестирования;
- поддержка системой машин различных архитектур;
- запуск системы в chroot-окружении.

Обработка аварийного завершения СТОС

Система тестирования, до настоящего времени находящаяся в эксплуатации, предполагала постоянное присутствие оператора, отслеживающего легко наблюдающиеся зависания на тесте, сбои в ядре и аппаратуре. Система автоматизированного тестирования прекращать работу в таких ситуациях не должна – все они должны быть отражены в отчетах оператору, а значит, корректно обработаны в Autotest.

Было выделено два типа ситуаций, которые могут послужить причиной аварийного завершения:

1. Зависание на одном из тестов.
2. Сбой в работе ядра (kernel panic) клиента, а также сбои в аппаратуре, повлекшие за собой аварийную перезагрузку машины.

Для борьбы с зависаниями системы тестирования была введена система таймаутов на каждый пакет тестов. По истечении таймаута пакету посылается сигнал *SIGTERM* и тестирование возобновляется со следующего пакета. Для корректного сбора результатов при аварийном завершении в соответствующие управляющие скрипты были добавлены обработчики этих сигналов, совершающие копирование результатов перед выходом. В список возможных результатов системы и каждого пакета тестов в отдельности

наряду с результатами *PASS* и *FAIL* был добавлен *TIMEOUT*.

Для анализа причины аварийной перезагрузки машины в архиве результатов сохраняются системные логи. По ним в дальнейшем легко определить конкретный тест, вызвавший *kernel panic*, и передать ошибку разработчику (вместе со стеком ядра и другой отладочной информацией). Корректно обрабатывается и полное зависание машины: в таких случаях, пользуясь интерфейсом устройства удаленного управления питанием, система пытается автоматически восстановить работоспособность машины. Если это сделать не удается, тестирование прекращается, а оператору приходит отчет о ситуации.

Поддержка различных архитектур

Хотя запуск системы выглядит на всех машинах единообразно, у архитектур «Эльбрус», SPARC и Intel x86 различаются загрузчики ОС, а значит, и способы установки и выбора конкретного ядра ОС. Загрузчиком, используемым архитектурой Intel, является *grub*, тогда как у архитектур «Эльбрус» и SPARC эту роль выполняет программа начального старта (ПНС).

Для поддержки загрузчика *grub* в Autotest был создан сервис *boottool*. Используя утилиту командной строки *grubby*, сервис позволяет выбирать загружаемое при следующем запуске ядро ОС путем модификации конфигурационного файла *grub*.

Для поддержки загрузчика «Эльбрус» и SPARC был написан аналогичный сервис *oslboottool*. Он позволяет загружать данные из файла конфигурации загрузчика во внутренний класс *oslbootconf*, менять в этом классе загружаемое по умолчанию ядро ОС и сохранять *oslbootconf* в формате файла конфигурации загрузчика.

Запуск СТОС в chroot-окружении

Для тестирования новой версии дистрибутива ОС без создания нового раздела файловой системы была поддержана возможность запуска СТОС в chroot-окружении. Для этого в автоматическом режиме производится монтирование системных каталогов, а также каталога результатов Autotest в chroot-директорию, после чего происходит смена корневого каталога. Далее запуск СТОС и сбор результатов происходят аналогично обычному запуску из корневого каталога. Директория для chroot может быть задана вручную оператором либо определена в Autotest автоматически.

Реализация

Система автоматизированного тестирования была реализована путем добавления в оригинальный Autotest контрольных файлов и соответствующих им тестов как со стороны сервера, так и со стороны клиента. Отдельно описаны интерфейсы работы с файлом конфигурации загрузчика ОС – *oslboottool*. Ниже приведен список добавленных контрольных файлов и их функции:

- Контроль версий – *OSL Check Archive*. На данном этапе происходит проверка архива сборок ОС на наличие непроверенных версий. Для каждой такой версии вызываются *OSL Install* и *OSL Verify*. Архитектура каждой тестовой машины также определяется на этом этапе – она передается в *Install* и *Verify* в качестве аргумента.
- Установка компонентов, включающих обновления ядра, модулей и самой системы тестирования ОС – *OSL Install*.
- Тестирование каждого ядра, составление и отправка отчета оператору – *OSL Verify*.

Для каждого непроверенного ядра новой версии осуществляются обработка файла конфигурации загрузчика и загрузка машины с тестируемым ядром, смена корневого каталога, запуск системы тестирования ОС с обработкой любых аварийных ситуаций, копирование результатов системы тестирования (или логов при *kernel panic*) в архив результатов. На этапе копирования происходят составление и отправка отчета оператору, в котором содержится информация об имени тестовой машины и ее архитектуре, а также результате тестирования каждого ядра данной версии (*PASS, FAIL, TIMEOUT, PANIC*). Кроме того, в приложениях к отчету для каждой проверенной версии ядра содержатся результаты каждого пакета тестов в отдельности, а также общий список всех невыполненных тестов.

Схема процесса автоматизированного тестирования на основе Autotest представлена на рис. 2.

Пользуясь интерфейсом командной строки Autotest, планировщик cron периодически ставит задачу *OSL Check Archive*, проверяющую архив сборок ОСОПО на наличие новых, непроверенных версий. Если таких версий обнаруживается несколько, они ставятся в очередь на проверку.

Ядра с различными конфигурациями последовательно проверяются на каждой машине закрытой подсети, причем тестирование на этих машинах происходит параллельно. После завершения тестирования его результаты и системные логи сохраняются в архиве результатов, доступном по сети. Оператору приходят отчеты о тестировании каждой версии ОСОПО на каждой машине.

Помимо автоматического тестирования операционной системы реализовано и другое применение системы: с помощью графического интерфейса Autotest Web Frontend указанная оператором стабильная версия программного обеспечения может быть установлена на любую машину с ОС «Эльбрус», доступную по сети АО «МЦСТ». Это

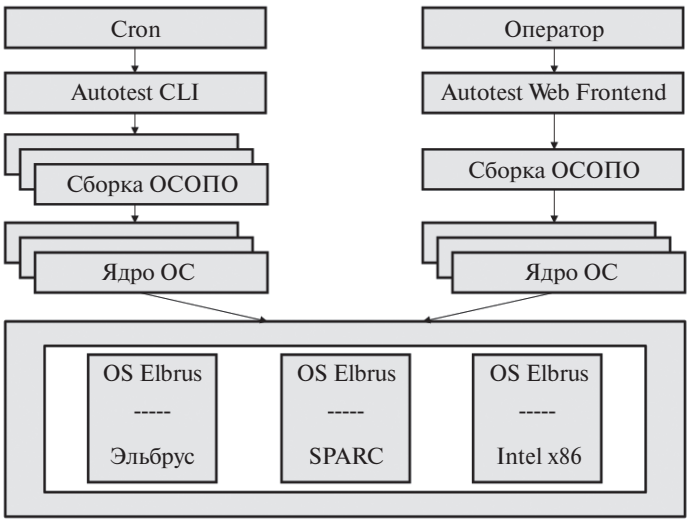

Рисунок 2. Схема процесса автоматизированного тестирования ОС

необходимо для тестирования новых аппаратных проектов.

Перспективы развития системы

Хотя базовая функциональность для автоматизации тестирования ОС «Эльбрус» была реализована, использование Autotest в качестве основы системы открывает новые возможности для дальнейшего развития:

- расширение базы тестов – добавление тестов для отслеживания регрессии производительности и стрессовых тестов;
- разработку и реализацию средств по графическому отображению результатов тестирования, позволяющих построить наглядные графики изменения показателей производительности;
- реализацию уровня сетевого взаимодействия по технологии Remote Command Call.

СПИСОК ЛИТЕРАТУРЫ

1. Bligh M., Whitcroft A. P. Fully Automated Testing of the Linux Kernel. In Linux Symposium, 2006, vol. 1, pp. 113–125.
2. Admanski J., Howard S. Autotest – Testing the Untestable. In Linux Symposium, 2009, pp. 9–18.

ИНФОРМАЦИЯ ОБ АВТОРЕ

Рыбаков Степан Андреевич, инженер-программист, АО «МЦСТ», 119334, Москва, ул. Вавилова, д. 24, тел.: 8 (916) 126-40-31, e-mail: stepan.a.rybakov@mcst.ru.

For citation: Rybakov S. A. Automation of the OS «Elbrus» testing based on the Autotest framework. Voprosy radioelektroniki, 2017, no. 3, pp. 52–56.

S. A. Rybakov

AUTOMATION OF THE OS «ELBRUS» TESTING BASED ON THE AUTOTEST FRAMEWORK

This article represents the automated testing system of the OS «Elbrus» kernel. The system is based on the open source project Autotest, implemented and developed by a number of leading international corporations. New functionality and solutions, implemented in Autotest during the system development, greatly contribute to the regular version upgrading of OS «Elbrus», based on OS Linux and created by leading MCST developers for original and standard high-performance platforms.

Keywords: OS Elbrus test cases, automated testing, Autotest.

REFERENCES

1. Bligh M., Whitcroft A. P. Fully Automated Testing of the Linux Kernel. In Linux Symposium, 2006, vol. 1, pp. 113–125.
2. Admanski J., Howard S. Autotest – Testing the Untestable. In Linux Symposium, 2009, pp. 9–18.

AUTHOR

Rybakov Stepan, software engineer, JSC «MCST», 24, Vavilova st., Moscow, 119334, Russian Federation, tel.: +7 (916) 126-40-31, e-mail: stepan.a.rybakov@mcst.ru.