

# Открытые системы

№01

2017

ISSN 1028-7493

ИТ для бизнеса —  
архитекторам  
информационных систем

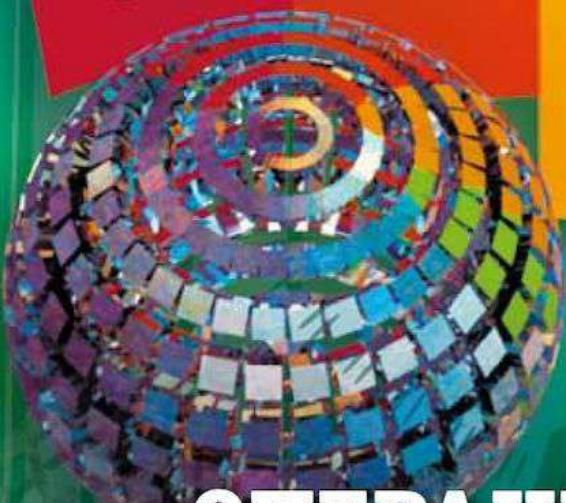
[www.osmag.ru](http://www.osmag.ru)

СУД

Открыты для вас. **25 ЛЕТ**

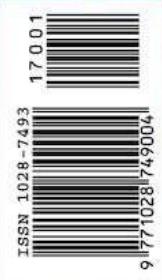


information.  
ally unique id.  
+ UUID= as a m  
dded and rem  
ant point?  
ct /  
host /  
var /var  
/dev/mapper/vg\_root\_lv /root  
/dev/mapper/vg\_root\_lv\_var /var  
/dev/mapper/vg\_root\_lv\_log /var/log  
/dev/mapper/vg\_root\_lv\_home /home  
EOF



## СОВРЕМЕННЫЕ ОПЕРАЦИОННЫЕ СИСТЕМЫ

- Запрещено все, что не разрешено
- Российская ОС уровня предприятия
- Платформа для встраиваемого ПО
- Интернет боевых вещей
- Что мешает миграции на Open Source?



**29** марта  
2017 года

[www.osp.ru/bigdata](http://www.osp.ru/bigdata)

# BIG'17 DATA

Шестой Российской форум

**BIG DATA 2017** – главное событие года для всех, кто хочет узнать о последних тенденциях, актуальных задачах и передовом опыте в области Больших Данных и продвинутой аналитики для бизнеса

**BIG DATA 2017** – это:

- ◆ опыт работы с передовыми аналитическими решениями от специалистов-практиков российского бизнеса и госуправления
- ◆ анализ рынка от экспертов ведущих международных фирм
- ◆ рассказ об инновациях от лидеров мирового рынка Большых Данных
- ◆ дискуссии и неформальное общение
- ◆ BIG DATA Expo



По вопросам участия: Ольга Пуркина



+7 495 725 47 80



[kon@osp.ru](mailto:kon@osp.ru)



[www.ospcon.ru](http://www.ospcon.ru)



'17

Организатор



**ОТКРЫТЫЕ  
СИСТЕМЫ**  
Open Systems Publications

# ОС эпохи Интернета вещей

**Ж**урнал «Открытые системы.СУБД» вступил в 25-й год своей истории, стартовав как издание для специалистов по операционным системам клона UNIX, первый номер этого года также посвящен ОС.

Рынок Интернета вещей к 2020 году достигнет 1,46 трлн долл., а по данным IDC, только европейские компании уже сегодня инвестируют почти 12 млрд долл. в соответствующее оборудование, ПО и сервисы, причем в ближайшие три года эти расходы будут ежегодно расти на 20%. Почему, однако, согласно различным опросам, почти 70% руководителей ИТ-служб уверены, что рост бизнеса в эпоху цифровой трансформации, катализатором которой во многом является Интернет вещей, обнаружит слабость традиционных ИТ-инфраструктур?

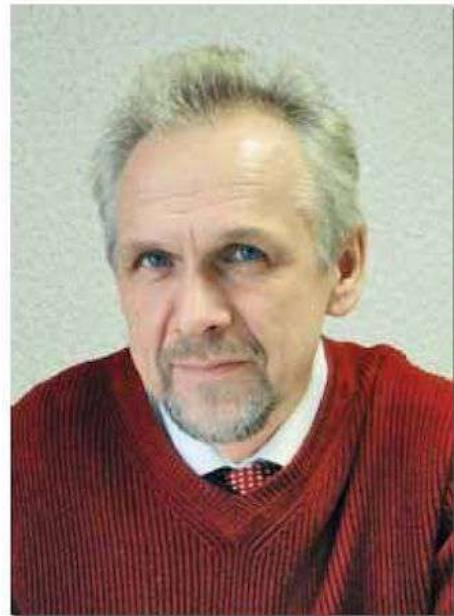
Действительно, появляются все новые устройства, призванные избавить человека от рутинных функций и выполняющие различные производственные или социальные задачи, но при этом сами они проще не становятся, усложняется и их программное обеспечение. К тому же эти устройства все чаще решают задачи сообща, а значит, кроме управления одним устройством, надо наладить эффективную работу их группировок. Неудивительно, что, наряду с блокчейном, умными роботами и дронами, платформы Интернета вещей, призванные предоставить удобные и надежные средства разработки для обеспечения безопасного функционирования многочисленных устройств, находятся сейчас на пике популярности «кривой хайпа» Gartner.

Неотъемлемой частью платформ Интернета вещей являются операционные системы, от которых в новых условиях требуются гарантия надежной поддержки выполнения встраиваемых приложений, безопасное управление индустриальными сетями из PLC-контроллеров и SCADA-систем, управление «умными» автомобилями и бесчисленным множеством других вещей. Однако при создании большинства существующих ОС не были предусмотрены средства защиты — соответствующие инструменты повышения их надежности интегрировались при необходимости в виде дополнительных модулей и функций, что лишь отчасти позволяло устра-

нить уязвимости. В условиях Интернета вещей нужны другие архитектурные решения, в которых изначально уделялось бы самое серьезное внимание вопросам безопасности. Приложения, выполняемые в таких средах, при любых условиях делают только то, для чего они предназначены, и будут защищены не только от внешних воздействий, но и друг от друга. В России недавно появилась такая ОС — это KasperskyOS, безопасность в которой предусмотрена на уровне архитектуры.

Другой аспект платформ Интернета вещей — управление в реальном времени группировками устройств, однако в классических операционных системах реального времени, как правило, таких возможностей нет. В основу еще одной отечественной системы, ОС РВ МАКС, положены принципы организации взаимодействия между несколькими устройствами, призванные упростить выполнение ряда типичных операций, возникающих в распределенных системах: распаралеливание при совместном решении какой-либо задачи, доступ к разделяемым данным, резервирование устройств и т. п. Принятая в этой ОС концепция распределенной общей памяти позволяет упростить программирование параллельно функционирующих устройств — разработчику надо лишь обеспечить выполнение действий с памятью, а ОС возьмет на себя заботу о согласованности данных на всех узлах системы, организует обмен данными и самостоятельно справится с возможными сбоями устройств или каналов связи. Понятно, что при работе группы устройств особое внимание должно быть уделено вопросам безопасности: ядро ОС РВ существует как собственные, так и доступные на целевом устройстве аппаратные средства защиты. Например, если в процессоре имеется поддержка уровней привилегий, то код ядра ОС всегда будет выполняться в привилегированном режиме, а код приложения — в обычном.

ОС для Интернета вещей должны обеспечивать самостоятельность устройства, работающего тем не менее в общем контексте, — каждая «вещь» получает и обновляет данные из общего контекста, не заботясь о наличии других, что дает ряд преимуществ. Например, имея доступ к контексту, сохраняющему текущий ста-



тус задачи, вновь подключаемое устройство продолжит выполнение задачи именно с того состояния, на котором неисправное прекратило свою работу, что обеспечивает надежность и масштабирование. Все это, как оказывается, уже используется военными, проводящими сегодня исследования по созданию Интернета боевых вещей (Internet of Battle Things, IoTB). Автономная адаптация сети и ее реорганизация без вмешательства человека, высокая плотность гетерогенных устройств (порядка миллиона вещей на квадратный километр), целенаправленные атаки на самые разные объекты, нарушение конфиденциальности, целостности и доступности информации путем электронной прослушки и внедрения вредоносов — все это будет скоро актуально и для гражданского Интернета вещей.

К 2018 году 22 млрд устройств смогут общаться между собой под управлением 200 тыс. новых приложений и сервисов, и при этом 66% гражданских сетей, используемых для поддержки Интернета вещей, будут иметь бреши в системах безопасности. Согласно опросу, проведенному аналитиками IBM Chief Information Security Officer, более 80% руководителей отделов информационной безопасности считают, что количество угроз будет расти, причем 60% из них сходятся во мнении, что их предприятия находятся сейчас в проигрышном положении. Внедрение новейших технологий, включая современные ОС, которые на уровне архитектуры будут решать проблемы надежного функционирования устройств Интернета вещей, становится в этих условиях главным приоритетом.

Дмитрий Волков

# ОТКРЫТИЕ СИСТЕМЫ. СУБД

Журнал основан в 1993 году

Главный редактор  
Дмитрий Волков, с.н.с., ИПМ РАН

Научный редактор  
Наталья Дубова

Редакционный совет:

Валерий Аджиев, к.т.н., с.н.с.,  
Национальный центр компьютерной анимации,  
Университет Борнхута (Великобритания);

Фуад Алекскеров, д.т.н., профессор, НИУ ВШЭ;

Михаил Горбунов-Посадов, д.физ.-мат.н.,  
зав. отделом ИПМ РАН, доцент, МГУ;

Юрий Зеленков, д.т.н., зав. кафедрой прикладной  
информатики, Финансовый университет  
при Правительстве РФ;

Сергей Д. Кузнецов, д.физ.-мат.н., профессор, МГУ;

Сергей О. Кузнецов, д.физ.-мат.н., профессор, НИУ ВШЭ;

Михаил Кузьминский, к.хим.н., с.н.с., ИХО РАН;

Александр Легалов, д.т.н., профессор, ОФУ;

Владимир Сухомлин, д.т.н., профессор, МГУ;

Павел Храмцов, к.т.н., доцент, МИФИ;

Игорь Федоров, к.т.н., профессор, МЭСИ;

Виктор Шнигман, д.т.н., профессор, МФТИ;

Леонид Эйсмонт, к.физ.-мат.н., научный  
консультант, НИИ «Квант»

Корректор Ирина Карпушина

Верстка и графика Мария Рыжкова

Дизайн обложки Денис Кирков

Адрес для корреспонденции:

127254, г. Москва, а/я 42

Телефоны:

+7 495 725-4780/84, +7 499 703-1854  
+7 495 725-4785 (распространение, подписка)

Факс: +7 495 725-4783

E-mail: osmag@osp.ru

Подписной индекс:

99482 — «Каталог российской прессы»

72733 — Объединенный каталог «Пресса России»

П2324 — Каталог ФГУП «Почта России»



© 2017 Издательство «Открытые системы»

Журнал зарегистрирован  
в Министерстве РФ по делам печати,  
телерадиовещания и средств массовых коммуникаций  
03.07.2015

Свидетельство ПИ № ФС 77-62328

Журнал выходит 4 раза в год

Цена свободная

Учредитель и издатель:

000 «Издательство «Открытые системы»  
Россия, 127254, Москва,  
проезд Добролюбова, дом 3, комн.13

Президент Михаил Борисов

Генеральный директор Галина Герасина

Директор ИТ-направления Павел Христов

Коммерческий директор Татьяна Филина

Все права защищены.

При использовании материалов  
необходимо разрешение редакции и авторов.

В номере использованы иллюстрации  
и фотографии: 000 «Издательство  
«Открытые системы» и IEEE Computer Society.

Отпечатано в ООО «Богородский  
полиграфический комбинат»  
142400, Московская область,  
г. Ногинск,  
ул. Индустриальная, д. 406  
(495) 783-9366, (49651) 73179

Тираж:  
4000 экз. — печатная версия,  
1062 экз. — PDF-версия

# Содержание № 1 (215) 2017

## НОВОСТИ. ФАКТЫ. ТЕНДЕНЦИИ.

Intel Atom переходит в высшую лигу

Mail.Ru строит частные облака на базе Tarantool

Amazon управляет облаком по ITIL

В Минообороне рассказали о новом суперкомпьютере

Accenture защитит блокчейн с помощью НСМ

Оптический диск на растительной основе

МВД получило «Эльбрусы»

Альянс во имя безопасности

Российскому рынку предложили

«суперкомпьютер по требованию»

Нейросети для диагностики рака

В МТИ приближают переход к кремниевой фотонике

R Server 9.0 преобразует модели в веб-сервисы

Oracle готовит разработчиков к Java 9

Облачное будущее разработки

Математики нашли способ сканивать Большие данные

Sixt предлагает Java-фреймворк

для создания микросервисов

## ПЛАТФОРМЫ

### 8 Китайский процессорно- суперкомпьютерный путь

Михаил Кузьминский

В конце 2016 года на Западе осознали, что  
пиковая производительность не только суперкомпьютеров, но и процессоров из Китая  
уже превзошла показатели подобных систем  
из США, — специалисты Поднебесной доказали,  
что могут обойти западные решения в об-  
ласти высокопроизводительных вычислений.

## В ФОКУСЕ: ОПЕРАЦИОННЫЕ СИСТЕМЫ

### 12 KasperskyOS: запрещено все, что не разрешено

Андрей Никишин

В 2016 году компания «Лаборатория Касперского»  
объявила о выводе на рынок своей операци-  
онной системы, предназначеннной для обеспе-  
чения безопасной работы сетевых устройств,  
защищенных не только от воздействий извне,  
но и друг от друга.

### 14 OS «Альт»: платформа уровня предприятия

Дмитрий Державин

Российская ОС «Альт» позволяет разверты-  
вать масштабируемые безопасные решения  
поддержки жизненного цикла корпоративных  
ИТ-инфраструктур.

### 16 Платформа для встраиваемого ПО

Александр Кучеров

Отечественная ОСРВ МАКС реализует необ-  
ходимый для работы встраиваемых систем  
функционал, позволяя организовать эффек-  
тивное взаимодействие устройств в распре-  
деленных системах.

### 19 OS и СУБД: мандатное разграничение доступа

Валерий Попов

Применение ОС Linux для различных приме-  
нений, требующих соблюдения конфиденци-  
альности, делает актуальной задачу обеспечен-  
ия безопасности связки ОС и СУБД, однако  
обеспечение сквозного мандатного разграни-  
чения доступа и независимости от платфор-  
мы — весьма сложная задача.

## БЕЗОПАСНОСТЬ

### 22 Риски пользовательских интерфейсов

Кристофер Хазард, Муниндар Сингх

Манипулирование — одно из самых вопиющих  
нарушений свободы личности, неизменно уни-  
жающее достоинство человека. Добывая иден-  
тифицирующие сведения, интеллектуальные  
пользовательские интерфейсы создают угрозу  
неприкосновенности личности и другие риски.

## ПРОГРАММНАЯ ИНЖЕНЕРИЯ

### 25 Создание критически важных

приложений на основе

микросервисов

Кристофф Фетцер

Ошибки в системном ПО непременно будут использоваться для атак, поэтому кри-  
тические приложения не должны зависеть от корректности ПО низкого уровня.  
Применение микросервисов и защищенных областей памяти обеспечивает требуемую  
надежность приложений без ущерба производительности.

## ИНТЕРНЕТ ВЕЩЕЙ

### 28 Интернет боевых вещей

Александр Котт, Анантрам Свами,

Брюс Вест

С расширением Интернета вещей входящим в него гражданским и военным системам по-  
требуются почти беспредельные способности  
масштабирования.

## ОПЫТ

### 31 Уроки Мюнхена:

миграция на свободное ПО

Марио Силич, Андреа Бэк

Стремительно растут темпы применения ПО с открытым кодом в различных областях, но в крупных организациях этот рост пока идет медленно.

## МИР

### 34 Обучение программированию в эпоху технологических революций

Виктор Иванников

В ИТ происходит перманентная революция: изменяются технологии, инструменты, появляются принципиально новые решения. Как в этих условиях организовать процесс обучения, подготовить грамотных программистов, опирающихся на фундаментальные знания при выполнении прикладных разработок?

### 40 Все дело в памяти

Наталья Дубова

Технологии, обеспечивающие высокую скро-  
стость обработки и масштабируемость хране-  
ния данных в памяти, не так просты, как ка-  
жется, хотя бы по причине разнообразия  
задач и данных.

## ИТ-УНИВЕРСИТЕТЫ

### 43 Кросс-языковая идентификация

авторов публикаций

Зинаида Апанович

Идентификация авторов публикаций важна для определения их научного рейтинга, од-  
нако при обработке имен русскоязычных ав-  
торов в англоязычных публикациях нередки ошибки. Система на основе комбинированно-  
го сравнения атрибутов и текстовых данных  
позволяет точно идентифицировать русско-  
язычных авторов.

## БИБЛИОТЕКА

### 46 Полвека на переднем крае ИТ

Александр Тыренко

Темы декабрьского номера 2016 года, январ-  
ского и февраля 2017 года жур-  
нала Computer (IEEE Computer Society, Vol. 49,  
No. 12, 2016 и Vol. 50, No. 1, 2, 2017) — взаимо-  
действие компьютерных устройств, будущее  
мира ИТ и технологии, дополняющие возмож-  
ности человека.

# ОТКРЫТЫЕ СИСТЕМЫ СЕГОДНЯ

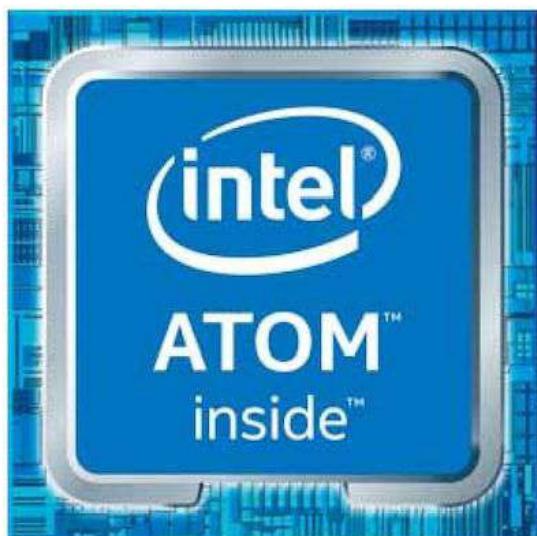
## Intel Atom переходит в высшую лигу

Похоже, Intel Atom больше не маломощный процессор для нетбуков и мобильных устройств: модели линейки C3000, которую анонсировали в корпорации, имеют до 16 ядер и особенности, присущие серверным чипам, в том числе сетевые функции и средства поддержки виртуализации. Процессоры Atom C3000 предназначены для массивов хранения, сетевого оборудования и устройств Интернета вещей. В дополнение к чипам в Intel предлагают комплекты разработчика, помогающие в создании ПО для сетевых устройств. Примечательная особенность C3000 — механизм автоматической коррекции ошибок,

ранее в основном применявшийся в Xeon старшего класса. Данная функция призвана защитить сетевое оборудование и системы хранения данных от сбоев. Atom C3000 приходят на смену линейке C2000, изначально предназначавшейся для микросерверов. Недавно в них был обнаружен дефект, способный вызывать сбои серверов и сетевых устройств. Корпорация также представила процессоры Xeon D-1500 для сетевого оборудования и систем хранения, рассчитанных на более высокую пропускную способность при обмене данными. Эти чипы имеют встроенный контроллер 10-Gigabit Ethernet и технологию QuickAssist, ускоряющую операции шифрования и компрессии данных.

## Mail.Ru строит частные облака на базе Tarantool

Компании Mail.Ru, IBS и Mellanox создают интегрированное решение для СУБД Tarantool с применением технологий линейки СКАЛА от IBS InterLab и сетевых технологий Mellanox. Решение позволит российским компаниям строить гибкие архитектуры ЦОД и будет интересно как крупным компаниям, так и организациям сектора государственного управления, заинтересованным в сведении информации о событиях по клиентам, продуктам, услугам, материалам, товарным запасам в единый пул с возможностью алгоритмического доступа ко всему массиву в реальном времени и в оперативном внедрении новых алгоритмов. Другим перспективным направлением применения решения может стать использование его в качестве хаба для Промышленного интернета: распределенная программная платформа Tarantool IIoT позволяет собирать данные с датчиков на производственных площадках, транспорте, сельскохозяйственных полях и передавать в ЦОД для анализа. Платформа обеспечивает скорость работы до 10–50 тыс. транзакций в секунду.





## Amazon управляет облаком по ITIL

У Amazon Web Services появился продукт Managed Services — набор сервисов управления работой инфраструктуры, в том числе администрирования и обеспечения безопасности облачной среды заказчика в AWS. Новые сервисы, адресованные наиболее крупным клиентам, планирующим переносить рабочие задачи в публичное облако, предлагает AWS совместно с ее партнерами. Инструменты, предоставляемые в рамках AWS Managed Services, помогают создавать автоматизированные процессы переноса унаследованных приложений в облако и последующего управления ими. Как сообщают в AWS, сервисы созданы на основе практик ITIL. Среди функций, реализуемых новым сервисом, — управление изменениями (создание автоматизированного процесса утверждения изменений, вносимых в среду AWS), управление инцидентами (для поиска причин и устранения ошибок), управление предоставлением ресурсов (для быстрого развертывания заранее подготовленных стеков инфраструктурных компонентов или приложений), управление заплатами, управление доступом, безопасность (антивирусная защита, распознавание и предотвращение вторжений), резервное копирование и отчетность (в том числе ведение подробных журналов операций для аудита).

## В Минобороны рассказали о новом суперкомпьютере

Министр обороны РФ Сергей Шойгу в эфире телеканала «Россия 24» рассказал о суперкомпьютере, установленном в Национальном центре управления обороной России, способном прогнозировать развитие вооруженных конфликтов. Суммарная вычислительная мощность суперкомпьютера составляет 16 PFLOPS, а максимальный объем данных, которые он способен хранить, — 236 Пбайт, сообщило агентство «Интерфакс».

## OCPB для распределенных систем

Российская компания «АстроСофт» объявила о создании операционной системы реального времени МАКС («МультиАгентные Когерентные Системы»), обеспечивающей разработку встраиваемого программного обеспечения для устройств Интернета вещей. Новая ОСРВ не только реализует классический функционал систем данного типа, например гарантированное время реакции на события внешней среды, но и обеспечивает надежное взаимодействие практически неограниченного количества устройств, а также встраивание внешних систем защиты данных. Сейчас ведутся пилотные проекты по применению МАКС для управления группировками расходомеров в ЖКХ и поддержки телеметрии линий электропередач. ОСРВ может найти применение в системах «Умный город» и решениях по обеспечению кибербезопасности отечественных систем различного назначения. ОС МАКС — это инициативная разработка «АстроСофта», на которую было потрачено два года без использования внешних инвестиций. В данный момент идет сертификация ОСРВ в ФСТЭК.

## Accenture защитит блокчейн с помощью HSM

В Accenture предлагают защищать распределенные реестры путем хранения ключей шифрования, используемых для заверения транзакций, в аппаратных криптографических модулях (hardware security module, HSM). Обычно HSM применяются банками для хранения PIN-кодов платежных карт и верительных данных, используемых в межбанковских платежах по системе SWIFT. В таком модуле данные защищены гораздо надежнее, чем при хранении на соединенных с сетями серверах, откуда информация может быть похищена в результате атаки. Защищаемые данные никогда не покидают HSM, и работа с ними жестко контролируется. Похожий механизм — «защищенный анклав» — есть также в iPhone, начиная с модели 5s. Концептуальная версия защитной системы Accenture работает с HSM-модулями nShield компании Thales e-Security и ПО распределенного реестра Hyperledger Fabric. В дальнейшем планируется обеспечить поддержку других HSM и блокчейнов. В сентябре прошлого года специалисты Accenture показали, как можно модифицировать транзакции в закрытом (permissioned) блокчейне, если у атакующего есть соответствующие верительные данные, — именно для защиты от подобных атак компания и предлагает свое решение. В открытых (permissionless) блокчейнах возможности провести такую атаку нет.

## Оптический диск на растительной основе

В холдинге «Росэлектроника» разработали оптический диск, состоящий в том числе из веществ растительного происхождения. Многослойные 3D-оптические носители включают в себя фоточувствительные слои, куда записываются данные, и полимерные, служащие для распределения светачитывающего лазера. Количество слоев колеблется от 40 до 60. Для формирования функциональных фоточувствительных слоев использованы вещества класса хромонов — фенольные соединения, образующиеся в растениях. В устройстве, разработанном специалистами ЦНИТИ «Техномаш», входящего в холдинг, запись производится за счет способности хромонов менять свои свойства под воздействием света определенной силы — они начинают светиться. Такой диск может содержать много больше записывающих слоев. Новый носитель может записывать данные объемом до 1 Тбайт при скорости чтения 12 Гбит/с. Для сравнения: стандартный диск Blu-Ray может содержать до 50 Гбайт при скорости считывания 576 Мбит/с.



## МВД получило «Эльбрусы»

Объединенная приборостроительная корпорация, входящая в состав «Ростеха», поставила в государственные ведомства первую партию серверов на процессорах «Эльбрус». Серверы используются миграционными службами и некоторыми другими подразделениями МВД. По заказу Минкомсвязи на базе отечественных серверов развернута часть инфраструктуры миграционных служб по проекту модернизации государственной системы изготовления, оформления и контроля паспортно-визовых документов нового поколения. Также отечественная техника использована для построения ЦОД федеральной информационной системы биометрических учетов и оперативно-розыскных данных МВД РФ. Одна из задач, решаемых новым ЦОД, — ускоренное проведение дактилоскопического анализа для повышения уровня раскрываемости преступлений. Весь цикл разработки — от архитектуры микропроцессора до топологии печатных плат и программного обеспечения — осуществляется в России. На платформе «Эльбрус» развернута отечественная операционная система с интегрированными средствами защиты от несанкционированного доступа. В рамках модернизации инфраструктуры паспортно-визовых служб на платформу «Эльбрус» перенесли программное обеспечение серверов баз данных, системы хранения данных, серверов приложений, веб-серверов, средства резервного копирования и мониторинга, средства IP-телефонии, сервис технической поддержки и множество других программных пакетов. На сегодняшний день это самый сложный проект, в котором используется серверное оборудование с микропроцессорами «Эльбрус».

## Альянс во имя безопасности

AT&T, IBM и Nokia образовали альянс IoT Cybersecurity Alliance, в который помимо инициаторов вошли компании Symantec и Palo Alto Networks, а также Trustonic, занимающаяся разработкой средств безопасности для мобильных устройств. Задача альянса — проведение исследований, обучение потребителей и коммерческих пользователей и оказание влияния на выработку стандартов и правил, касающихся безопасности Интернета вещей. По мере распространения технологий Интернета вещей повышается риск появления новых уязвимых мест, что стало очевидным в прошлом году после массированных DDoS-атак, в которых были использованы взломанные видеокамеры систем наблюдения и другие устройства. Однако спектр потенциальных уязвимостей гораздо шире и охватывает уровни сети, облачных систем и приложений. По данным AT&T, за последние три года число попыток сканирования устройств Интернета вещей на уязвимости выросло более чем в 30 раз.

## Искусственный интеллект предсказывает действия толпы

Сотрудники Института научно-исследовательских компьютерных технологий университета Санкт-Петербургского национального исследовательского университета информационных технологий, механики и оптики совместно с Национальным автономным университетом Мексики, Амстердамским университетом, Массачусетским технологическим институтом и Северо-Восточным университетом в Бостоне разработали программу PULSE для моделирования и прогнозирования поведения толпы. Например, что будет при пожаре или наводнении или что произойдет при блокировке одного из выходов из помещения — в каких местах наиболее вероятна давка. Основная задача PULSE — повышение безопасности при проведении глобальных массовых мероприятий.



## В Китае создают прототип экзафлопсного суперкомпьютера

Появление рабочего суперкомпьютера мощностью в один экзафлопс и выше, а также приложений для него можно ждать не раньше 2020 года, однако прототип такой системы в Китае предполагается разработать уже в этом году, заявил инженер Национального суперкомпьютерного центра в Тяньцзине на заседании местного собрания народных представителей. Экзафлопсный суперкомпьютер будет в 200 раз мощнее первого китайского петафлопсного суперкомпьютера Tianhe-1, ставшего в 2010 году самым мощным в мире. В США запуск экзафлопсного суперкомпьютера запланирован только на 2023 год. Есть опасения, что новая администрация уменьшит финансирование работ в области высокопроизводительных вычислений. Например, сайт The Hill утверждает, что их планируется снизить до уровня 2008 года — такую рекомендацию дал аналитический центр The Heritage Foundation.

## Российскому рынку предложили «суперкомпьютер по требованию»

Принципиально новую для ИТ-рынка услугу — «суперкомпьютер по требованию» — запустили бывшие сотрудники Parallels, IBM и МФТИ. Предложенный одноименной российской компанией HPC Hub прошел стадию бета-тестирования и начинает работать с клиентами. HPC Hub представляет собой облачную суперкомпьютерную платформу по требованию (HPC as a Service, HPCaaS) для решения задач научно-исследовательской деятельности с предустановленным расчетным ПО. Это позволяет сэкономить десятки миллионов рублей на покупке и обслуживании суперкомпьютера и на лицензиях коммерческого ПО. Речь идет о предоставления удаленного доступа к полноценному виртуальному суперкомпьютеру, что отличает сервис HPC Hub от сервисов Amazon Web Services, Microsoft Azure и Google Cloud Platform, предоставляющим в основном доступ к изолированным виртуальным машинам. Клиентами HPC Hub уже стали: НПО «Союзнефтегазсервис», для которого создан гибридный виртуальный HPC-кластер и предоставлена услуга посutoчного билинга; лаборатория биоинформатики ФНКЦ физико-химической медицины, которая использовала HPC Hub для анализа метагеномных данных; ряд видеостудий. Первый прототип HPC Hub был разработан в 2012 году, а активная разработка начата в 2015 году, когда к проекту подключился первый инвестор — Дмитрий Михайлов, генеральный директор EG Capital Partners.

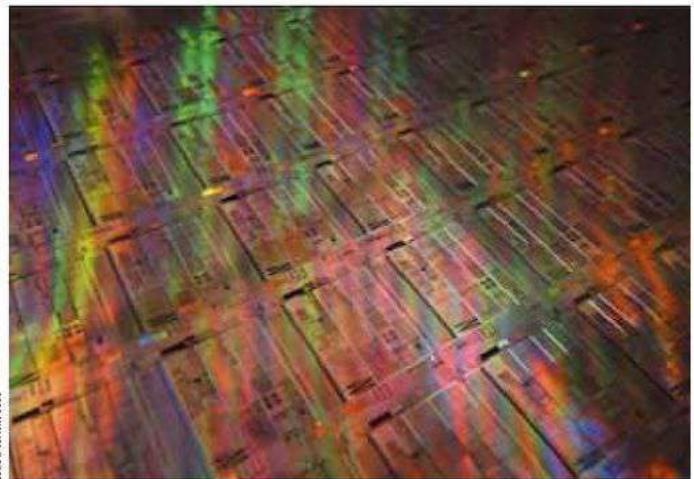
# НОВОСТИ. ФАКТЫ. ТЕНДЕНЦИИ.

## Нейросети для диагностики рака

Департамент информационных технологий Москвы совместно со столичным Департаментом здравоохранения разрабатывает технологию на основе нейронных сетей, позволяющую распознавать рак легких. Тесты показали, что такая технология корректно распознает опухоли в 97% случаев. Нейронная сеть, «натренированная» на распознавание новообразований, разработана подразделением ДИТ, отвечающим за использование в городском хозяйстве результатов анализа Больших Данных. Эксперимент стал частью пилотного проекта мэрии по раннему выявлению рака легких с помощью низкодозной компьютерной томографии. Отмечается, что диагноз по-прежнему будет ставить врач, однако нейронная сеть поможет уточнить диагностику. В будущем планируется использовать нейросети для проверки всех результатов компьютерной томографии для постановки точного диагноза.

## В МТИ приближают переход к кремниевой фотонике

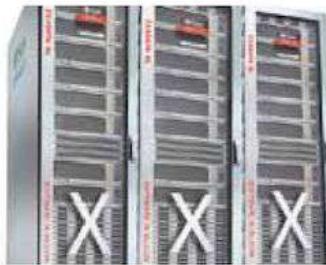
По оценкам участников полупроводниковой индустрии, при нынешних темпах роста потребностей компьютеров в электричестве к 2040 году для них уже будет недостаточно всей электроэнергии, вырабатываемой в мире. Потребляемую мощность микросхем можно было бы резко уменьшить, если вместо электронов для переноса данных пользоваться фотонами. За последние 20 лет был достигнут значительный прогресс в развитии кремниевой фотоники — оптических элементов, легко интегрируемых с традиционными микросхемами. Но в таких элементах используются иные физические принципы, нежели в оборудовании волоконно-оптических сетей. В частности, в последних применяются нелинейные явления второго порядка, за счет чего повышаются КПД и надежность обработки оптических сигналов. Исследователи из МТИ опубликовали доклад, посвященный разработанному ими методу использования нелинейностей второго порядка в кремниевой фотонике. Сообщается также о создании прототипов двух кремниевых устройств, использующих такие явления, — модулятора, кодирующего данные в оптическом пучке, и удвоителя частоты, компонента, важного для разработки лазеров, которые можно точно настраивать на различные частоты. Как отметили в IBM, достижение ученых МТИ сыграет важную роль в дальнейшем развитии кремниевой фотоники, где прогресс задерживала прежняя невозможность использования нелинейных эффектов второго порядка.



Источник: МТИ

## Первая Exadata на SPARC

Корпорация Oracle представила Exadata SL6 — новую версию своей интегрированной системы Exadata, построенную на RISC-процессорах SPARC M7. До сих пор все системы этой линейки были основаны на серверах стандартной архитектуры на процессорах Intel Xeon. За этим исключением конфигурация Exadata SL6 мало отличается от выпущенной ранее системы Exadata X6-2. Обе системы оптимизированы для обслуживания баз данных Oracle и используют операционную систему Oracle Linux, межсоединения InfiniBand и узлы хранения данных на базе жестких дисков и флеш-накопителей. Выпущенный в 2015 году процессор SPARC M7 состоит из 32 ядер и работает на тактовой частоте 4,1 ГГц. Кристалл этого процессора оборудован аппаратным ускорителем Data Analytics Accelerator для более быстрой обработки аналитических запросов к базе данных. В этом году в Oracle собираются представить новый процессор SPARC S7 с полностью интегрированными компонентами. Путем объединения двухсокетных вычислительных узлов системы в кластер Exadata SL6 в максимальной конфигурации масштабируется до 20 вычислительных узлов с 640 процессорными ядрами и 10 Тбайт оперативной памяти.



Источник: Oracle

## R Server 9.0 преобразует модели в веб-сервисы

В Microsoft продолжают пропагандировать исследователям данных (data scientist) достоинства применения функционального языка программирования R: компания анонсировала Microsoft R Server 9.0, платформу выполнения аналитических скриптов R. Пользователи смогут создавать модели на R, размещать их в SQL Server и превращать в веб-сервисы, что упрощает использование результатов анализа данных в приложениях, способных, например, прогнозировать задержки вылетов или помогающих оптимизировать рекламные кампании. Функция создания сервисов будет доступна как в платном варианте Microsoft R Server, так и в открытом дистрибутиве Microsoft R Open. В компании также представили пакет функций машинного обучения для Microsoft R Server 9.0, MicrosoftML, доступный на Windows и SQL Server, а в дальнейшем и на Linux и Hadoop.

## Oracle готовит разработчиков к Java 9

При подготовке к июльскому выпуску Java 9, в Oracle опубликовали JDK 9 Migration Guide — руководство по переводу приложений на новую версию платформы. Код, использующий только официальный API платформы Java SE, должен работать без изменений, но в Oracle не дают аналогичных обещаний по поводу программ, которые обращаются к определенным внутренним функциям, советуя разработчикам, планирующим перенос приложений, ознакомиться с ранней сборкой Java 9 и обновить сторонние библиотеки. В новой версии применяется более совершенный механизм сборки мусора, отсутствует компонент JavaDB, созданный на базе движка СУБД Apache Derby, исключена клиентская виртуальная машина для платформы Win32 — остался только серверный вариант, обладающий, как утверждается, более высоким быстродействием, но и более ресурсоемкий. Исключены также некоторые средства для macOS, в частности движок AppleScript.

## Облачное будущее разработки

По убеждению главы компании Heroku Адама Гросса, благодаря облачным платформам PaaS воплотится в жизнь идея о том, что все без исключения коммерческие компании станут разработчиками ПО. Выступая на конференции для разработчиков, он заявил, что даже компании, не занимавшиеся разработкой, должны научиться управлять клиентскими приложениями «не хуже Amazon, Facebook и Google», — это станет доступным благодаря возможностям быстрой разработки и развертывания, обеспечиваемым PaaS. Он отметил, что в последние два-три года произошло разделение инструментов разработки на применяемые и администрируемые локально и внешние, управляемые операторами облачных платформ. По прогнозу Гросса, со временем программисты полностью перейдут на облачные сервисы, вследствие чего локальное программное обеспечение для разработки ПО исчезнет. Он также напомнил о том, что платформы PaaS предоставляют собственную инфраструктуру безопасности для разрабатываемых приложений, сервисы работы с данными на основе популярных проектов с открытым кодом, возможности контейнеризации и создания систем на основе микросервисов.

## Математики нашли способ сжимать Большие Данные

Ученые Лаборатории искусственного интеллекта МТИ и Хайфского университета разработали метод поиска подмножеств, сохраняющих ключевые математические отношения своих источников — огромных срезов данных. Способ отличается универсальностью и применимостью в широком круге областей, включая анализ текстов на естественном языке, машинное зрение, обработку сигналов, системы выдачи рекомендаций, прогнозирование погоды, финансовую аналитику, нейробиологию и др. Метод основан на геометрической интерпретации данных — представлении их в виде гиперсферы и поиске средних значений в подмножествах. Репрезентативность выбранных подмножеств исследователи доказывают математически. Действуя по принципу понижения размерности, метод позволяет радикально уменьшить затраты на анализ разреженных данных с помощью широко применяемых методик, таких как латентно-семантический анализ, метод главных компонент и др. Ученые показали действенность своего метода на примере матрицы, устанавливающей соответствие между статьями англоязычной «Википедии» и используемыми в них словами. Такая таблица содержит 1,4 млн строк (статьей) и 4,4 млн столбцов (слов). Алгоритм позволил выявить кластеры слов, наиболее характерных для 100 самых распространенных тем в «Википедии». Например, кластер со словами «платье», «невеста», «подружка» и «свадьба» соответствует теме свадеб, а со словами «оружие», «выстрел», «заклинил», «пистолет» и «стрельба» — теме стрельбы.

## Sixt предлагает Java-фреймворк для создания микросервисов

Обеспечение возможности создания микросервисов на Java — одно из приоритетных направлений работы Oracle, однако в компании Sixt, предоставляющей услуги проката автомобилей, независимо от Oracle разработали фреймворк, реализующий те же возможности, — Java-micro. Он позволяет создавать сервисы в форме контейнеров Docker или файлов fat jar. Фреймворк имеет подключаемый реестр сервисов и позволяет организовать обработку событий с помощью платформы Apache Kafka. Предусмотрен интерфейс вызова функций других сервисов

## Блокчейн не роскошь

Издательство «Открытые системы» провело международную конференцию «Технологии блокчейна: платформы, архитектуры и опыт реальных проектов», на которой рассматривались решения и проекты, потенциал которых выходит далеко за рамки криптовалют. Выступления продемонстрировали: блокчейн, в основе которого лежат принципы прозрачности взаимодействия процессов и коллегиального владения данными, способен обеспечить повышение эффективности бизнеса и многих государственных структур. В работе конференции приняли участие более 150 представителей компаний из различных отраслей. «Время вопросов, быть или не быть блокчейну, прошло. Пора заняться обсуждением конкретных проектов и конкретных платформ», — заявил Дмитрий Волков, руководитель программного комитета конференции.

На пленарной сессии обсуждались технологические основы и экосистема блокчейна, проблемы безопасности, перспективы использования распределенных реестров и пилотные проекты в финансовой сфере, возможности применения блокчейна в существующих и новых моделях бизнеса.

Во время работы параллельных тематических сессий участники конференции познакомились с платформами и архитектурами блокчейна, а также с примерами практического использования технологии. В их числе: аккредитивы «Альфа-банка»; платформа «Мастерчейн», представленная в докладе Татьяны Бильк, эксперта Банка России; торговые площадки на базе смарт-контрактов; страховой онлайн-агрегатор Prosto.Insure.

Конференция продолжила цикл мероприятий издательства «Открытые системы», посвященных технологиям баз данных и Больших Данных, а учитывая популярность тематики практического применения распределенных реестров (см. журнал «Открытые системы. СУБД» № 4, 2016), такие конференции планируется сделать регулярными.



и обработки ошибок. Java-micro поддерживает ведение журнала операций в формате JSON, выдачу отчетов о производительности и балансировку нагрузки на стороне клиента, имеет механизм переноса баз данных. Java-micro разрабатывается с расчетом на поддержку Java, языка Google Go и распространяется по лицензии Apache. Незадолго до выхода фреймворка организация Eclipse Foundation взяла под свое крыло еще один проект подобного рода — MicroProfile. Независимые проекты в области микросервисов появились в связи с сомнениями в стремлении Oracle активно развивать Java для предприятий.

# Китайский процессорно-суперкомпьютерный путь

В конце 2016 года на Западе осознали, что пиковая производительность не только суперкомпьютеров, но и процессоров из Китая уже превзошла показатели подобных систем из США, — специалисты Поднебесной доказали, что могут обойти западные решения в области высокопроизводительных вычислений. Кроме этого, выпустив собственные процессоры, имеющие производительность на уровне графических ускорителей и построив на их платформе быстродействующие системы, китайцы, вероятно, стали первоходцами решений только на гомогенных узлах, не содержащих дополнительных акселераторов.

Ключевые слова: суперкомпьютер  
Keywords: HPC, ShenWei, Sunway TaihuLight, Supercomputer



Михаил Кузьминский

Суперкомпьютер Sunway BlueLight, ставший одним из результатов программы 863 [1], еще в конце 2011 года занял 14-е место в Top500 и 39-е — в рейтинге энергоэффективных высокопроизводительных вычислительных систем Green500. Этот суперкомпьютер, в отличие от ряда других массово-параллельных систем, был построен на базе узлов с разработанными и производимыми в Китае 64-разрядными RISC-микропроцессорами SW1600 третьего поколения семейства

ShenWei, созданного в КНР в основном для военного применения. В конце 2016 года китайцы уже не просто заняли два первых места в ноябрьском списке Top500, обогнав по производительности системы из США, но и продемонстрировали мировому сообществу суперкомпьютер Sunway TaihuLight на базе собственных 260-ядерных процессоров четвертого поколения SW26010. Данная система стала первым в мире суперкомпьютером с пиковой производительностью свыше 100 PFLOPS и заняла при этом 4-е место в Green500, обойдя системы на базе Intel

Xeon Phi x200 (Knights Landing, KNL) [2]. На тесте HPGC суперкомпьютер Sunway TaihuLight занял 4-е место. Теперь на него обратил внимание и Джек Донгарра [3], один из составителей рейтинга Top500.

Главное достижение четвертого поколения ShenWei — это оригинальная архитектура процессора SW26010, который, как и Intel KNL, содержит большое число процессорных ядер и имеет близкую к старшей модели KNL пиковую производительность (чуть больше 3 TFLOPS, см. таблицу) выполнения операций с плавающей запятой двойной точности (DP). Как и KNL,

SW26010 позволяет создавать гомогенные суперкомпьютеры (без акселераторов), хотя сами по себе многоядерные SW26010 гетерогенны, а не гомогенны, как KNL.

Процессор SW26010 (рис. 1) [4, 5] имеет 260 ядер и включает четыре группы ядер CG (Core Group). Каждая группа содержит кластер из 64 вычислительных элементов (Computing Processing Element, CPE), которые и образуют основу вычислительной мощности процессора. Кроме CPE, связанных в кластере решеткой-массивом  $8 \times 8$ , каждая группа CG имеет одно свое ядро общего назначения — процессорный элемент управления (Management Processing Element, MPE). Процессор SW26010 имеет тактовую частоту 1,45 ГГц, однако по какой технологии он изготовлен, точных сведений нет: Донгарра указывает на 28 нм (<https://science.energy.gov/~media/ascr/ascac/pdf/meetings/201609/Dongarra-ascac-sunway.pdf>), однако в [5] говорит о 40 нм.

Элементы CPE и MPE имеют 64-разрядную RISC-архитектуру с поддержкой 256-разрядных векторов (по четыре числа двойной точности, вдвое короче, чем у KNL) и команд «умножить и сложить». Каждый MPE содержит два векторных конвейера, соответственно, все 256 CPE дают  $256 \times 8$  DP-результатов за такт и еще четыре MPE дают  $4 \times 8 \times 2$  результатов. Если умножить это на 1,45 ГГц, получается, что суммарная пиковая производительность SW26010 составляет 3062 GFLOPS.

Кроме процессорных ядер CPE и MPE, каждая CG содержит свой работающий с DDR3 контроллер памяти (Memory Controller, MC), а объединяются CG через общую для SW26010 «сеть на микросхеме» (Network on Chip, NoC). Процессор поддерживает связь с другими устройствами через системный интерфейс SI (рис. 1).

## Процессоры Xeon Phi KNL и SW26010

	Xeon Phi 7290 (KNL)	ShenWei SW26010
Число ядер	72	260
Тактовая частота, ГГц	1,5	1,45
DP-результатов за такт	32	16 в MPE, 8 в CPE
Пиковая производительность (GFLOPS)	2995	3062
Производительность (GFLOPS) на Watt на тесте Linpack	5,0 <sup>1</sup>	6,1 <sup>2</sup>
Высокоскоростная память	MCDRAM 16 Гбайт, 490 Гбайт/с <sup>3</sup>	Нет (но есть сверхоперативная)
Главная память	DDR4/2400, до 384 Гбайт, 90 Гбайт/с <sup>4</sup>	DDR3/2133, 32 Гбайт <sup>4</sup> , 136 Гбайт/с
Арифметическая интенсивность, FLOPS/байт	6,1	22,4

<sup>1</sup> На суперкомпьютере Oakforest-PACS с серверами Fujitsu PRIMERGY CX1640 M1 с Xeon Phi 7250.

<sup>2</sup> На суперкомпьютере Sunway TaihuLight...

<sup>3</sup> Пропускная способность в тесте stream/triad.

<sup>4</sup> Емкость в узле Sunway TaihuLight, пиковая пропускная способность.

Разработчики говорят об «общей глубоко объединенной многоядерной архитектуре» (Deeply Fused Many-Core, DFMC). Число CPE и топология их соединения в других DFMC-процессорах могут отличаться, а топология NoC — это решетка, колыцо или перекрестный коммутатор (crossbar), и она может подбираться в зависимости от количества компонентов в DFMC (число CG, MPE и MC может меняться [5]). Поэтому DFMC можно охарактеризовать как высокомасштабируемую гетерогенную микроархитектуру, ориентированную на высокопроизводительные вычисления.

Каждая группа CG имеет собственное адресное пространство памяти, связывающей ядро MPE и кластер CPE через контроллер MC. Ядра CPE и MPE имеют совместимую 64-разрядную RISC-архитектуру, напоминающую DEC Alpha [5], образованную из 212 команд, работающих с 8-, 16-, 32- и 64-разрядными операндами. Кроме того, CPE поддерживают коммуникации на уровне регистров и передачи потоков данных, а также синхронизацию в их класс-

тере. Суперскалярный уровень в MPE поддерживает декодирование сразу четырех команд и возможность параллельного выполнения семи команд, а в CPE декодируются и выполняются параллельно по две команды [6].

В DFMC память разделяется между MPE и CPE, а когерентность кэша достигается только между MPE. В MPE имеется двухуровневый кэш: кэш первого уровня команд (I) и данных (D) емкостью по 32 Кбайт и общий кэш второго уровня для команд и данных (256 Кбайт). В этих кэшах используются строки размером 128 байт. Кэши первого уровня в MPE являются четырехканальными наборно-ассоциативными, а кэш второго уровня — восьмиканальным. В CPE устроено по-другому: здесь имеются кэш команд первого уровня емкостью 16 Кбайт и сверхоперативная память SPM (scratch pad memory) емкостью 64 Кбайт с NUMA-задержками. Кроме того, в кластере CPE предусмотрен еще общий кэш второго уровня для команд [5]. В отличие от кэша, сверхоперативная память

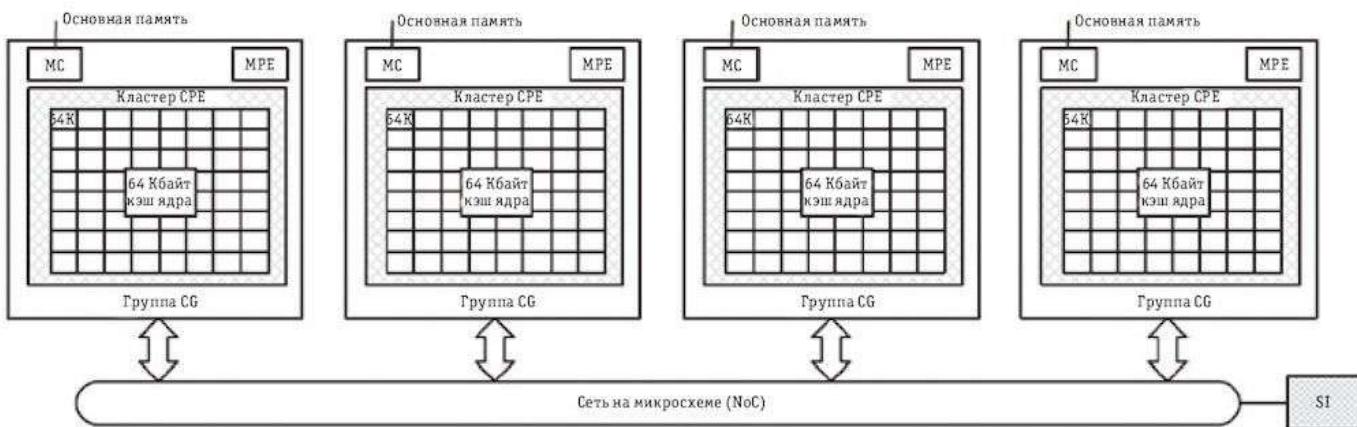


Рис. 1. Архитектура SW26010

# платформы

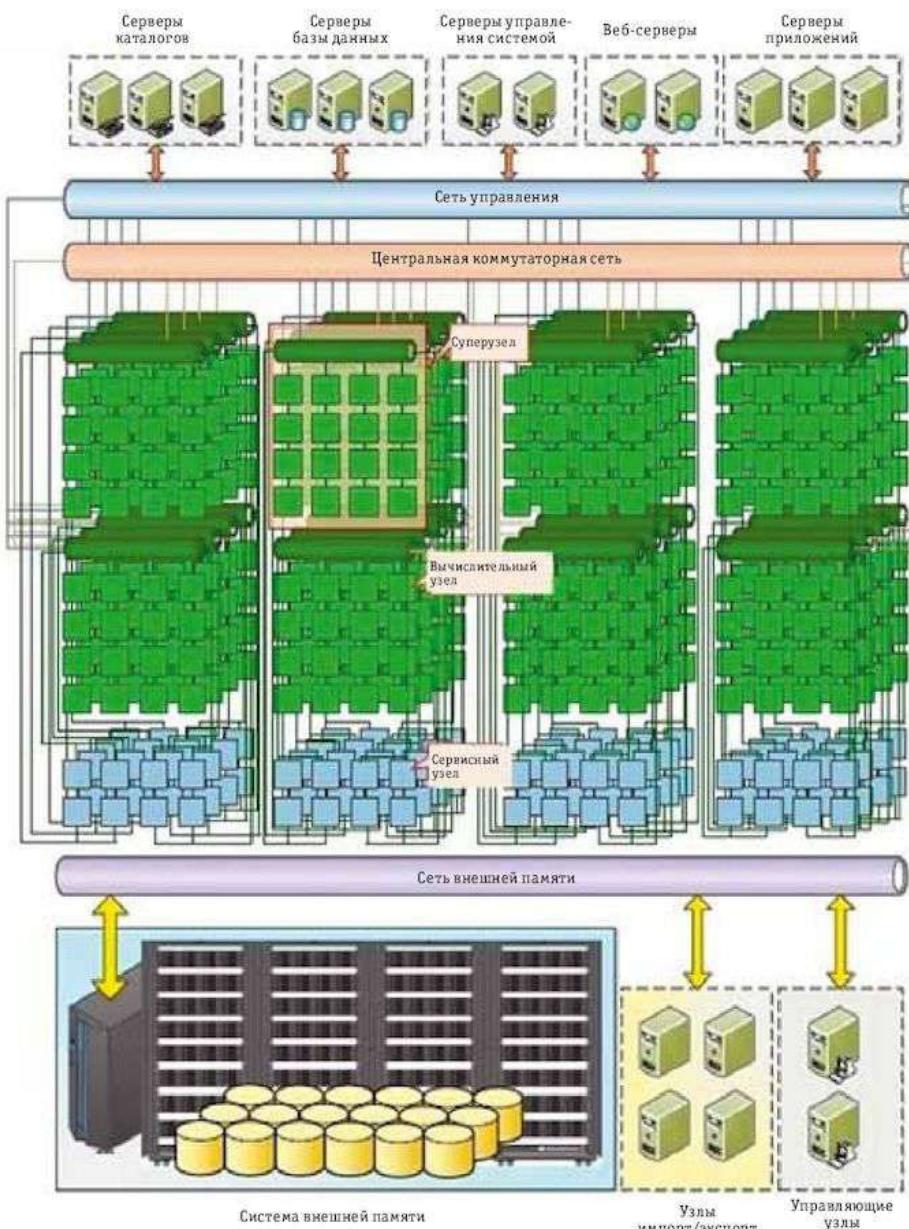


Рис. 2. Общая архитектура Sunway TaihuLight

не содержит копию данных из основной памяти и может быть сконфигурирована как быстрый буфер, напрямую управляемый пользователем, или как программино-эмулируемый кэш, который дает автоматическое кэширование данных, что, естественно, сильно уменьшает производительность. Здесь усматривается некая аналогия со сверхоперативной памятью MCDRAM в KNL. Выбор SPM по сравнению с D-кэшем позволяет упростить аппаратную реализацию.

Слабое место SW26010 — иерархия памяти. Процессоры Intel при работе с современной памятью DDR4 поддерживают более высокую пропускную способность, чем SW26010. По размерам буфера и кэшей CPE-клUSTERы уступают не только дос-

тупным на рынке GPU и MIC-ускорителям Intel (в том числе KNL), но и процессорам Xeon E5-26XX v4 и v3. Поэтому важнейшей при оптимизации программ для SW26010 задачей является уменьшение нагрузки на память, особенно в CPE-клUSTERе.

Полнофункциональное ядро MPE может работать в системном или пользовательском режимах, поддерживая функции прерывания и управления памятью и реализуя внеочередное спекулятивное выполнение команд. На MPE эффективно выполняются последовательные части программ.

Ядра CPE работают только в пользовательском режиме, имеют ограниченные функции и не поддерживают прерывания. В них реализовано статическое предска-

зание переходов и поддерживаются команды с плавающей запятой типа «умножить и сложить» и деление / извлечение квадратного корня. CPE и MPE могут работать независимо, а SPM оснащен собственным адресным пространством, поэтому в CPE имеются команды для обмена данными между SPM и основной памятью. В CG есть общая память, но архитектура DFMC ориентируется на «кооперативную» технику, и CPE аппаратно поддерживают синхронизацию и межъядерные коммуникации, в частности передачу потоков данных и коммуникации на уровне регистров. При доступе в основную память, включая передачу потоков данных, CPE могут получить копию из кэша MPE.

Для взаимодействия ядер CPE внутри CG имеется аппаратура сети CPE\_NET, которая и обеспечивает, например, коммуникации регистров и интерфейс с NoC.

Ядра MPE, занимая небольшую часть площади процессора и потребляя малую долю электроэнергии, предназначены для повышения общей производительности, а на CPE построена упрощенная микроархитектура, позволяющая получить высокую производительность для распараллеленных приложений с плавающей запятой. Для программиста независимость MPE от CPE означает, что можно сделать программу для выполнения на MPE или на кластере CPE. В целом, скорее всего, для оптимизации программ на SW26010 требуется больше ресурсов на программирование, чем в случае KNL.

Процессор SW26010 использует технологию систем на кристалле: кроме контроллеров MC, там интегрированы интерфейсы PCIe 3.0, Gigabit Ethernet и стандарт JTAG для тестирования и отладки микросхем.

Суперкомпьютер Sunway TaihuLight включает шесть систем: вычислительную, сетевую, внешней памяти, обслуживания и диагностики, электропитания и охлаждения, программного обеспечения (рис. 2) — и представляет собой иерархическую систему с общей пиковой производительностью 125 PFLOPS (93 PFLOPS на Linpack HPL в Top500) на базе однопроцессорных вычислительных узлов, имеющих память по 32 Гбайт (по 8 Гбайт на CG). Два вычислительных узла образуют карту, четыре карты — плату, а из 32 плат собирается суперузел на 256 вычислительных узлов. Из четырех суперузлов состоит кабинет, а весь Sunway TaihuLight состоит из 40 кабинетов. Суперузел имеет полно связанный скрещиваемый коммутатор для поддержки вычислительно интенсивных работ.

Сетевая система суперкомпьютера трехуровневая: на верхнем находится центральная коммутаторная сеть, через которую работают все суперузлы; на среднем — суперузловая сеть; на нижнем — сеть совместного использования ресурсов, связывающая ресурсы с суперузлами и обеспечивающая службы коммуникаций ввода-вывода и отказоустойчивости вычислительных узлов. Бисекционная полоса пропускания в Sunway TaihuLight составляет 70 Гбайт/с, пропускная способность канала сети — 16 Гбайт/с, а диаметр сети (наибольшее число переходов между узлами) равен семи.

Система внешней памяти включает сеть внешней памяти, подсистему управления и дисковый массив с общей памятью 20 Пбайт, имеющий системную консоль, управляющий сервер и сеть для управления всей системой. Донгарра в докладе в министерстве энергетики США указывал на наличие в Sunway TaihuLight 288 SSD-дисков емкостью по 800 Гбайт и общую пропускную способность ввода-вывода на уровне 288 Гбайт/с.

Система обслуживания и диагностики отвечает за интерактивное управление узлами, суперузлами и вычислительной системой в целом — мониторинг состояния Sunway TaihuLight, обнаружение сбоев, ведение протоколов и т. п. Охлаждение вычислительной и сетевой систем основано на непрямом водяном охлаждении, а в системе внешней памяти применяется смешанное воздушно-водяное охлаждение.

Система программного обеспечения Sunway TaihuLight базируется на Linux-подобной операционной системе Raise OS 2.0.5 с компиляторами для языков Фортран, Си/С++ с расширениями Sunway OpenACC (применяется OpenACC 2.0 со специальными дополнениями для Sunway TaihuLight; OpenACC 2.0 поддерживается в компиляторе GCC 6). Для распараллеливания внутри CG применяется Sunway OpenACC, для четырех групп CG внутри процессора SW26010 можно применять распараллеливание на OpenMP или MPI, а между узлами — MPI. Обеспечиваются и базовые математические библиотеки, для чего применима библиотека xMath, сопоставимая с коммерческими библиотеками типа MKL и др. (это описано в указанном выше докладе в министерстве энергетики США).

Слабое место Sunway TaihuLight — недостаточно высокая пропускная способность памяти, обусловленная особенностями SW26010, а также не самые высокие параметры быстродействия межсоединения [3].

Как видно из списка тестов HPCG, по показателю отношения производительности к пиковому значению Sunway TaihuLight уступает всем суперкомпьютерам, кроме системы «Ломоносов».

В [4] указано, что Sunway TaihuLight ориентирован не только на традиционные НРС-вычисления, но и на обработку больших массивов данных. Первая доступная информация об ожидаемой производительности относилась к FPGA-моделированию SW26010. В Национальном суперкомпьютерном центре в Уси, где установлен Sunway TaihuLight, выполняется множество различных приложений, включая расчеты атмосферных моделей и приложения вычислительной

гидродинамики, молекулярной динамики и др., для которых приводятся данные о достигнутой производительности. Однако там нет информации по эталонным программам, широко используемым в НРС. Появились только данные для GTC-P/PIG с использованием на нижнем уровне OpenACC, а выше — MPI.

В [6] приводится подробное описание оптимизации на SW26010 широко применяемого в вычислительной гидродинамике программного комплекса OpenFOAM, написанного на С++. Эта оптимизация включала применение коммуникации регистров в кластере СРЕ, векторизацию и др. Однако на кластере СРЕ поддерживается только среда на Си. Кроме того, для OpenFOAM особенно важна пропускная способность памяти: например, на кластере СРЕ оптимизированный код выполняется в восемь раз быстрее, чем на МРЕ, и на 18% быстрее, чем на одном процессорном ядре Intel Xeon E5-2695 v3 / 2,3 ГГц. Таким образом, весьма вероятно, что для достижения высокой эффективности приложений на Sunway TaihuLight/SW26010 потребуется большая работа программистов.

\*\*\*

Специалисты Поднебесной уже доказали, что могут обойти США в области высокопроизводительных вычислительных систем, однако пока имеется отставание по высокоскоростной памяти. Нет открытых данных и по возможности организации широкомасштабного производства китайских процессоров и по их стоимости. Продемонстрированные на Sunway

TaihuLight данные о производительности приложений пока почти не относятся к эталонным, активно используемым во всем мире, что затрудняет сравнение. Вполне вероятно, что для оптимизации приложений еще потребуются значительные доработки — даже OpenACC используется не настолько часто, и работы для программистов здесь будет больше, чем в случае Intel KNL.

**Экзафлопсы против математического моделирования**  
Как превратить математическое моделирование в востребованный инструмент для получения новых фундаментальных знаний и высокотехнологичных новаций на основе нового поколения прикладных программ, работающих на компьютерах экстремальной производительности?

Валерий Ильин

Открытые системы. СУБД 2013, № 05

Появление процессоров Xeon Phi и ShenWei нового поколения, имеющих производительность и количество ядер, сравнимые с акселераторами, включая GPU, а также создание на их основе суперкомпьютеров, лидирующих в Топ500, — это отражение новой тенденции, которая выражается в ориентации на гомогенные узлы без применения дополнительных акселераторов. ■

## ЛИТЕРАТУРА

1. Дмитрий Волков. Стратегические ИТ: китайский сюрприз 863 // Открытые системы. СУБД. — 2010. — № 3. — С. 32–37. URL: <http://www.osp.ru/os/2010/03/13001879> (дата обращения: 18.03.2017).
2. Михаил Кузьминский. Из ускорителей в процессоры // Открытые системы. СУБД. — 2016. — № 3. — С. 26–28. URL: <http://www.osp.ru/os/2016/03/13050252/> (дата обращения: 17.03.2017).
3. Dongarra J. Report on the Sunway TaihuLight System // PDF. [www.netlib.org](http://www.netlib.org). Retrieved June. — 2016. — T. 20.
4. Fu H. et al. The Sunway TaihuLight supercomputer: system and applications // Science China Information Sciences. — 2016. — T. 59. — № 7. — C. 072001.
5. Zheng F. et al. Cooperative computing techniques for a deeply fused and heterogeneous many-core processor architecture // Journal of Computer Science and Technology. — 2015. — T. 30. — № 1. — C. 145–162.
6. Meng D. et al. Hybrid Implementation and Optimization of OpenFOAM on the SW26010 Many-core Processor. 2016. URL: [http://hpc.sjtu.edu.cn/hpcchina16\\_openfoam.pdf](http://hpc.sjtu.edu.cn/hpcchina16_openfoam.pdf) (дата обращения: 17.03.2017).

Михаил Кузьминский ([kus@free.net](mailto:kus@free.net)) — сотрудник, Институт органической химии РАН (Москва).

# KasperskyOS: запрещено все, что не разрешено



В 2016 году компания «Лаборатория Касперского» объявила о выводе на рынок своей операционной системы, предназначеннной для обеспечения безопасной работы сетевых устройств, защищенных не только от воздействий извне, но и друг от друга.

*Ключевые слова: безопасность, безопасные операционные системы  
Keywords: Security, Trusted Operating Systems*

Андрей Никишин

Работа над KasperskyOS началась в 2002 году с идеи о том, что индустрии требуется полноценная безопасная операционная система, обеспечивающая превентивную защиту от злоумышленников. В большинстве существующих ОС изначально не закладывались меры безопасности, поэтому соответствующие инструменты обеспечения надежности интегрировались в них в виде дополнительных модулей и функций, что не решало фундаментальной проблемы уязвимости. А что, если сделать операционную среду, в которой вообще будет невозможно каким-либо программам выполнять посторонние функции?

В среде, безопасность которой предусмотрена на уровне архитектуры, должно быть запрещено все, что заранее не было разрешено, тогда даже при обнаружении «дыр» злоумышленник не сможет ими воспользоваться. Иными словами, программы внутри такой системы при любых условиях делают только то, для чего они предназначены в соответствии с предварительно прописанными ограничениями.

В качестве теоретической базы разработчики KasperskyOS использовали книги «Радужной серии» [1]. В них, в частности, отмечается, что безопасность компьютерных систем никогда не будет идеальной и правильнее говорить не о безопасных, а о доверенных системах. В одной из книг описывается концепция доверенной вычислительной базы (*Trusted Computer Base, TCB*) — совокупности механизмов, реализующих политику безопасности и определяющих степень доверия к системе. Все без исключения взаимодействия происходят

только по указанию и с разрешения ТСВ, а основное правило — «запрещено все, что не разрешено».

KasperskyOS — микроядерная операционная система, в ядре которой (рис. 1) прописаны диспетчер процессов, механизм межпроцессного взаимодействия (inter-process communication, IPC) и система мониторинга обращений (reference monitor), получившая название Kaspersky Security System (KSS). Все остальные процессы и компоненты: управление памятью и периферийными устройствами, драйверы файловых систем и т. п. — в данном случае работали бы против концепции безопасности.

Процессы в ОС взаимодействуют между собой и с функциями ядра, отправляя и получая IPC-сообщения. Для каждого из этих сообщений KSS решает, разрешить (allow) его или запретить (deny). Применяется принцип default deny «по умолчанию»: если KSS не обнаруживает четкого правила, разрешающего то или иное действие, оно запрещается.

При написании приложений для Kaspersky OS используется особый подход — «компонентная модель», в основе которой лежит понятие «сущность» (entity). Сущностью может быть как целая программа, так и ее отдельная функция. Программа включает в себя набор компонентов со своими сущностями, и, таким образом, выполнение программы в Kaspersky OS превращается в IPC-переписку сущностей.

Чтобы участвовать в переписке, сущность обязана удовлетворять одному важному условию: в ней должен присутствовать код интерфейса доступа к механизму микроядра IPC. Нужно отметить, что разработчик не участвует в создании этой части кода,

код автоматически генерируется из описания интерфейса на IDL (Interface Definition Language) — C++-подобном языке спецификации интерфейсов. Строгая типизация IDL позволяет проводить формальную верификацию корректности взаимодействия одной сущности с другой и проверять код на безошибочность.

С помощью кода интерфейса формируются две функции: Proxy для клиентских приложений и Dispatch — для серверных. Клиентское приложение вызывает функцию серверного приложения или ядра системы, передает параметры функции Proxy и сериализует их (то есть упаковывает в формат IPC-сообщения). Затем приложение вызывает транспортную функцию IPC в микроядре, передает ей созданное IPC-сообщение, ждет ответного IPC-сообщения, десериализует его (распаковывает параметры для вызываемой функции) и передает сделавшему вызов базовому коду клиента. Функция Dispatch делает обратное: получает IPC-сообщение, десериализует его, передает параметры базовому коду связанного с интерфейсом сервиса и, наконец, сериализует результат в IPC-сообщение.

Если в сущности имеется много разных функций, то они описываются на языке CDL (Component Definition Language). Специально разработанный компилятор Nk генерирует единый в рамках компонента код с интерфейсом, который на самом деле представляет собой совокупность Dispatch-интерфейсов всех входящих функций.

Для описания многокомпонентных сущностей имеется язык EDL (Entity Definition Language), с помощью которого описываются также и отдельные функции с собственными Dispatch-интерфейсами. При

# в фокусе: операционные системы

компилировании EDL-файла формируется общий код сущности с единственным Dispatch-интерфейсом. Найти адресата для него можно по уникальному идентификатору Runtime Interface ID (RIID), который генерируется на этапе компиляции EDL-описания сущности. Такая вложенность типизированных спецификаций позволяет создавать сложные программы, в которых каждая функция будет снабжена собственным Proxy- или Dispatch-интерфейсом.

IPC-взаимодействие — это дело двух сущностей, в чем-то напоминающее технологию P2P, однако, в отличие от нее, происходящее по принципу randevu. Чтобы randevu состоялось, создается канал обмена IPC-сообщениями путем выделения сущностям глобальных системных дескрипторов (указателей, handle), идентифицирующих сущности отправителя и получателя. Как только сущности становятся владельцами своих дескрипторов, открывается IPC-канал. Каждая сущность знает только о выделенном ей дескрипторе, а об их паре знает только механизм IPC. Формирование IPC-канала называется «спариванием дескрипторов» (handles pairing). После спаривания посторонний участник не может вклиниваться в диалог, а канал остается открытый до тех пор, пока сущности остаются владельцами дескрипторов. Модель IPC-взаимодействия handles pairing запатентована «Лабораторией Касперского».

Никто не может вклиниться в IPC-диалог, однако система KSS может просматривать проходящие по каналу сообщения. В составе KSS выделяются две основные части:

- модуль Security Server, принимающий решение о вердикте на основе политики безопасности (рис. 2);
- структура Decision Cache, хранящая вердикты по отдельным политикам для повышения производительности перлюстрации.

Решение разрешать или запрещать IPC-взаимодействие принимается в соответствии с политикой безопасности, которая зависит от свойств и целей системы. Политика безопасности описывается с помощью формального аппарата — например, в терминах темпоральных логик. Чтобы связать конкретные действия сущностей с конкретными политиками безопасности, был разработан декларативный язык конфигураций безопасности CFG. Конфигурация безопасности, составленная на этом языке, в сочетании с IDL-описанием интерфейса сущности позволяет компилятору Nk сгенерировать структуру данных Gate с уникальным идентификатором SID (Security ID). Эта структура связывает сущность с по-



Рис. 1. Архитектура Kaspersky OS



Рис. 2. Принцип работы Kaspersky Security System

литикой безопасности, и если у сущности нет структуры Gate, то к ней применяется принцип default deny и она отбрасывается.

Багаж унаследованных приложений иногда не позволяет полностью заменить имеющуюся у пользователей ОС на KasperskyOS, поэтому в некоторых случаях достаточно внедрить KSS в уже существующую операционную систему.

Есть два способа создания приложений, работающих под управлением KasperskyOS:

- *Перенос существующих приложений, использующих POSIX API.* После переноса процессы, происходящие внутри такого ПО, не будут контролироваться ОС, однако его внешние связи будут безопасными.
- *Разработка новых приложений.* С точки зрения обеспечения безопасности это лучший вариант: функции будут проверены системой, и при написании кода программисту не надо задумываться о безопасности. Писать для операционной системы KasperskyOS не сложнее, чем для Linux, — здесь используется тот же API, а дополнительный инструментарий и языки, поставляемые в KasperskyOS SDK (компилятор Nk, IDL, EDL, CDL, CFG), ускоряют разработку безопасного ПО. Однако на этапе разработки

архитектуры требуется продумать разбиение на домены безопасности (сущности) и сформировать политику безопасности, которая будет реализована впоследствии параллельно с созданием функциональной части приложения.

\*\*\*

Система KasperskyOS предназначена для управления индустриальными сетями, телекоммуникационным оборудованием и устройствами Интернета вещей в целом. В планах развития ОС — разработка дополнительных функциональных моделей, упрощающих как применение системы, так и создание безопасных приложений. ■

## ЛИТЕРАТУРА

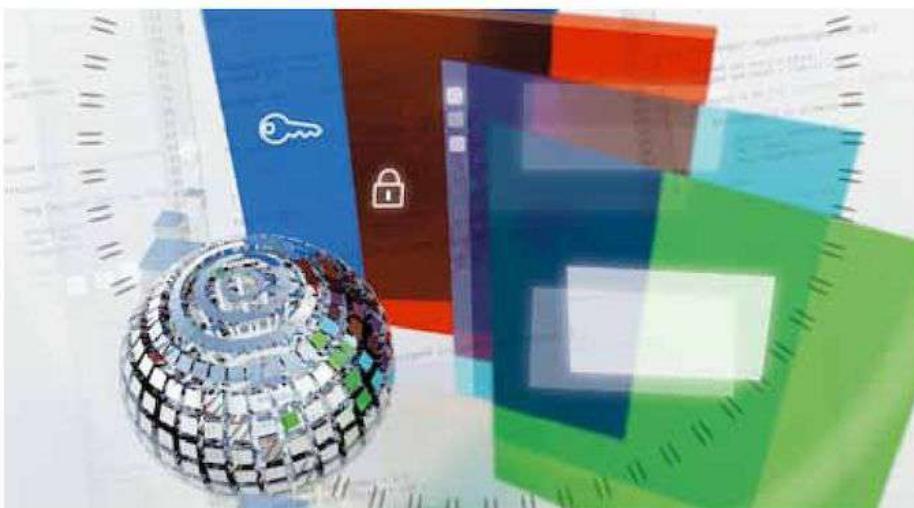
1. Руслан Богатырев. Защищенные операционные системы // Открытые системы. СУБД. — 2001. — № 4. — С. 39–43. URL: <http://www.osmag.ru/os/2001/04/180077> (дата обращения: 8.03.2017).

Андрей Никишин ([Andrey.nikishin@kaspersky.com](mailto:Andrey.nikishin@kaspersky.com)) — руководитель отдела развития технологических проектов, компания «Лаборатория Касперского» (Москва).

# ОС «Альт»: платформа уровня предприятия

Российская ОС «Альт» позволяет развертывать масштабируемые безопасные решения поддержки жизненного цикла корпоративных ИТ-инфраструктур.

*Ключевые слова:* «Базальт СПО», операционные системы, свободное ПО  
*Keywords:* Free Software, Operating Systems



Дмитрий Державин

Все продукты компании «Базальт СПО», в том числе ОС уровня предприятия «Альт Сервер» и «Альт Рабочая станция», построены на технологиях и архитектурных решениях, созданных и поддерживаемых командой разработчиков «Альт» (Alt Linux Team) и компанией «Альт Линукс», полноправным преемником которой теперь является «Базальт СПО».

Облик ОС «Альт» как самостоятельной операционной системы оформился в начале 2001 года и окончательно сложился в 2005 году с выходом версии 3.0. Предпосылкой к созданию ОС в то время была необходимость поддержки русскоязычных пользователей в части форматов, стандартов, протоколов и бизнес-процессов, востребованных на постсоветском пространстве. Проект был открыт изначально — предполагалось, что любой желающий мог присоединиться к команде и внести свой вклад в развитие ОС.

Все ОС семейства «Альт» выпускаются на базе полностью российской разработки — репозитория «Сизиф» [1], входящего

сегодня в число крупнейших самостоятельных репозиториев в мире. Поддержку репозитория, а также инфраструктуры сборки осуществляет компания «Базальт СПО». Помимо возможности использования оригинальных инструментов разработки команды «Альт» (репозитория исходного кода и бинарных пакетов, аннотированного индекса пакетной базы и системы отслеживания ошибок), участие в команде позволяет досконально изучить систему и избежать в будущем потенциально опасных ситуаций типа привязки к конкретному производителю или внедрения слабых технических решений.

Особое внимание в платформе «Альт» уделяется обеспечению безопасности построенных на ее основе систем, что достигается благодаря архитектуре и принятию таких упреждающих мер, как исправление уязвимостей до публикации кода в открытых источниках. Среди архитектурных решений можно отметить механизм изоляции приложений, подверженных сетевым атакам, и механизм управления фиксированными состояниями параметров системы, чувствительных к ошибкам администратора.

Многие решения, впервые внедренные командой «Альт», впоследствии были повторены участниками других крупных проектов разработки ОС, такими как Debian или RedHat. Среди этих решений, например, сборка бинарных пакетов в изолированном воспроизводимом окружении и сборка пакетов под непосредственным управлением системы контроля версий.

Сегодня ОС «Альт» — это технологическая платформа разработки, внедрения и поддержки программно-технических решений для широкого спектра аппаратных конфигураций. Платформа обеспечивает совместимость компонентов на уровне пользовательских и программных интерфейсов, а также совместимость интеграционных решений [2].

В области обработки данных ОС «Альт» предоставляет широкие возможности для офисного применения (оформление текстов, электронных таблиц, схем, презентаций), анализа и обработки мультимедийных данных, решения геоинформационных задач, в том числе с поддержкой клиентской части сервисов протоколов общепринятого семейства стандартов Open Geospatial Consortium (OGC), моделирования и проектирования.

В состав ОС «Альт» входят готовые к развертыванию популярные сетевые службы: доступ к графическим и текстовым терминалам с поддержкой «тонких» и «толстых» клиентов; возможность удаленной загрузки и централизованного хранения пользовательских конфигураций, доступа к файлам; протокол НТТР с поддержкой масштабирования и балансировки нагрузки; механизмы работы с СУБД разных типов. Для большинства сетевых сервисов поддерживается возможность аутентификации с единой точкой входа (SSO, Single Sign-On), а также средства мониторинга и событийного управления.

При выполнении проектов миграции ОС особое внимание приходится обращать на такие компоненты, как сервис каталогов, средства коллективной работы, платформа поддержки облаков, средства виртуализации, инструменты обеспечения безопасности, интерфейс администратора и инфраструктура разработки.

Домен безопасности ОС «Альт» основан на Samba DC — совместимой с Microsoft Active Directory реализации домена безопасности, предоставляющей единое пространство учетных записей пользователей и групп, централизованные настройки доступа к ресурсам и сквозную авторизацию в сети с единым входом. Поддерживается как серверная, так и клиентская часть: рабочие станции «Альт» могут входить в домен Active Directory на базе Windows Server, а рабочие станции Windows — в домен «Альт». Обеспечивается плавная миграция с домена Windows на домен «Альт».

Для организации совместной работы в корпоративной сети применяется открытое решение SOGo — сервис, который поддерживает обмен сообщениями, контактами, расписаниями событий, а также обеспечивает разграничение доступа, интеграцию с доменом Samba DC и поддержку клиентов на базе Windows. Для SOGo обеспечивается миграция с Microsoft Exchange Server.

Для поддержки облаков в ОС «Альт» интегрированы инструменты OpenStack [3], позволяющие автоматизировать управление физическими и виртуальными узлами, осуществлять мониторинг и балансировку нагрузки, а также организовать поддержку инфраструктуры виртуальных рабочих мест.

Средства управления виртуализацией в ОС «Альт» представлены инструментом ALT PVE, предназначенный для поддержки как «легких» изолированных окружений, так и полноценных виртуальных машин. Реализована также возможность организации виртуальных рабочих мест. Средства виртуализации в сертифицированных дистрибутивах ОС «Альт» входят в комплекс средств защиты информации.

В состав платформы «Альт» входит интерфейс администратора — Alterator,

применяемый для быстрой настройки компонентов ОС и включающий в себя три унифицированных пользовательских интерфейса: веб-, графический и интерфейс командной строки, — а также более сотни модулей конфигурирования, начиная от задания имени узла до настройки домена безопасности и системы резервного копирования. В Alterator реализован многопользовательский режим работы сгибким разграничением прав.

В дистрибутивах «Альт» поддерживаются актуальные версии сред выполнения приложений на языках C/C++, Java, Python, Perl, Ruby, PHP, JavaScript, Go и др. Для большинства языков в ОС «Альт» доступны инструменты (в том числе и интегрированные визуальные среды) поддержки полного цикла проектирования, разработки, отладки, тестирования и развертывания приложений, включая средства автоматизации процесса сборки и упаковки приложений в воспроизводимом изолированном окружении.

В дистрибутивах ОС «Альт» имеются прикладные библиотеки, помогающие в разработке программ для решения задач двумерной и трехмерной визуализации пространственных и геопространственных

данных, для анализа и обработки аудио-визуальных данных, создания клиентских и серверных сетевых масштабируемых и распределенных приложений. Кроме того, имеются библиотеки по методам математической статистики, численным методам, инструменты для выполнения символьных

## Надежные стены Linux

Из надежных дистрибутивов в России, пожалуй, наиболее известен ALT Linux, а на Западе в эту когорту входят Owl, Trusted Debian и Trustix Security Linux. Если есть несколько реализаций утилит или программы, то в дистрибутив попадает тот, который построен с большими требованиями к безопасности, даже если он обладает меньшей функциональностью.

Валерий Коржов

Открытые системы. СУБД, № 07–08

вычислений и моделирования физических объектов. Для многих прикладных задач реализована поддержка высокоуровневых программных каркасов, в том числе для быстрой разработки на специализированных языках и на скриптовых языках общего назначения.

Благодаря открытости, платформа позволяет обеспечить совместимость с программными комплексами сторонних производителей — например, с системами антивирусной и криптографической защиты, электронного документооборота, системами обеспечения финансово-хозяйственной деятельности и другим специализированным ПО от партнеров компании «Базальт СПО».

Системы на базе дистрибутивов «Альт» работают на аппаратных plataформах

x86 и x86-64, имеются сборки для ARM v7, ARM v8 («Байкал-М»), ведутся работы по переносу на платформу «Эльбрус», исследована возможность переноса на архитектуру MIPS («Байкал-T1»).

\*\*\*

Линейка ОС «Альт» включена в Реестр российских программ, и хотя формально эта процедура не отличается сложностью, фактически речь идет о появлении полноценной отечественной платформы поддержки жизненного цикла корпоративной ИТ-инфраструктуры. Среди основных направлений дальнейшего развития ОС «Альт» можно отметить разработку механизмов внедрения, расширение поддержки сертифицированных прикладных программных решений и аппаратных архитектур. ■

## ЛИТЕРАТУРА

1. Валерий Коржов. Во власти скриптов // Открытые системы. СУБД. — 2002. — № 9. — С. 48–50. URL: <https://www.osp.ru/os/2002/09/181934> (дата обращения: 18.03.2017).
2. Алексей Гриневич, Денис Марковцев, Владимир Рубанов. Проблемы совместимости Linux-систем // Открытые системы. СУБД. — 2007. — № 1. — С. 10–15. URL: <https://www.osp.ru/os/2007/01/3999198> (дата обращения: 17.03.2017).
3. Дмитрий Волков, Лев Левин. Секреты успеха OpenStack // Открытые системы. СУБД. — 2015. — № 2. — С. 14–17. URL: <http://www.osp.ru/os/2015/02/13046272> (дата обращения: 15.03.2017).

Дмитрий Державин (dd@basealt.ru) — ведущий инженер, компания «Базальт СПО» (Москва).

# Платформа для встраиваемого ПО



Отечественная ОСРВ МАКС реализует необходимый для работы встраиваемых систем функционал, позволяя не только ускорить разработку ПО для новых устройств на основе микроконтроллеров, но и организовать эффективное взаимодействие устройств в распределенных системах.

*Ключевые слова: ОС реального времени, управление распределенными инфраструктурами  
Keywords: management of distributed infrastructures, real-time operating system*

Александр Кучеров

**А**втоматизация все глубже вторгается в жизнь общества, появляются новые устройства, призванные избавить человека от рутинных задач, однако при этом сами устройства проще не становятся. Рост сложности микроэлектронных устройств приводит к усложнению их программной начинки, что, в свою очередь, порождает потребность в удобных, надежных и доступных средствах разработки. В полной мере это относится и к операционным системам, эффективность применения которых при создании встраиваемых приложений в конечном счете определяет время вывода на рынок нового продукта, уменьшение стоимости разработки, а также надежность функционирования устройств на основе микроконтроллеров.

Большинство встраиваемых систем предназначены для управления объектами или мониторинга параметров внешней среды, что накладывает жесткие ограничения на время реакции системы при обработке событий [1], поэтому одно из главных требований к встраиваемой ОС — обеспечение выполнения многозадачного приложения в условиях реального времени. Кроме того, немаловажными для встраиваемых систем являются компактность, автономность и безопасность, что означает минимизацию используемых для нужд ОС ресурсов, наличие режима энергосбережения, повышенные требования к скорости выполнения системных функций и высокую отказоустойчивость (обработку исключительных ситуаций и ошибок в приложении, контроль системного стека и стека задач). Когда речь идет о работе распределенной системы, состоящей из

множества устройств, то дополнительно к требованиям для отдельных устройств возникает задача организации структуры этой системы, мониторинга состояния каждого устройства, обеспечения надежной связи между ними, резервирования оборудования и т. д.

Сегодня на рынке встраиваемых ОС есть как открытые (FreeRTOS, eCos, RTEMS, ChibiOS), так и проприетарные решения ( $\mu$ C/OS, VxWorks, ThreadX, VDK) [2]. Однако, несмотря на разнообразие предложений на рынке, при интеграции ОС в конкретный проект разработчики часто сталкиваются со множеством проблем, в числе которых ограниченный функционал, отсутствие доступа к исходным кодам, несовместимость с аппаратной платформой, скучная документация, отсутствие технической поддержки, сложность программного интерфейса и т. д. Для разработки встраиваемых распределенных систем, ориентированных на отечественную элементную базу, российским разработчикам и пользователям нужна ОС, позволяющая упростить процесс разработки ПО и устраниТЬ возможные ограничения, связанные с применением западных программно-аппаратных платформ.

Проект ОСРВ МАКС (Операционная Система Реального Времени для МультиАгентных Когерентных Систем) стартовал в 2015 году, и его результатом стало то, что сегодня на рынке появилась система, не только отвечающая стандартным требованиям к ОС реального времени, но и обладающая дополнительными возможностями по обеспечению взаимодействия между устройствами в распределенной среде. Система поставляется с комплектом русскоязычной документации, поддерживает работу с оборудова-

нием отечественного производства (продукция «ПКК Миландр» — 32-разрядные микроконтроллеры серий 1986 и 1967), а ее техническая поддержка осуществляется российскими специалистами. Кроме того, к исходным кодам системы прилагаются шаблоны проектов для различных сред разработки и готовые программы, помогающие пользователям быстро освоить ОС, настраивать и создавать новые приложения.

В состав ядра системы (рис. 1), обеспечивающего работу многозадачного приложения, входят: планировщик (диспетчер), объекты синхронизации (семафоры, события и др.) и средства обеспечения взаимодействия задач (очереди сообщений). Планировщик поддерживает два режима многозадачности.

- **Кооперативная многозадачность** снижает накладные расходы работы системы за счет исключения лишних переключений задач и использования объектов синхронизации. Режим позволяет заранее предопределить последовательность выполнения задач, возлагая, однако, на разработчика ответственность за установку точек передачи управления между задачами.

- **Вытесняющая многозадачность** избавляет разработчика от необходимости планирования задач вручную (планирование с учетом точек и последовательности передачи управления между задачами) и обеспечивает оптимальную загрузку процессора.

При вытесняющей многозадачности планирование задач осуществляется на основании их приоритетов, что позволяет разработчику определить критичность каждой задачи в приложении — аппаратные ресурсы будут распределяться между задачами в зависимости от их критичности. В этом режиме поддерживается механизм

# в фокусе: операционные системы

предотвращения инверсии приоритетов (так называемое наследование приоритетов), исключающий возможность задержки выполнения высокоприоритетных задач при ожидании ресурса, захваченного задачей более низкого приоритета. Наконец, обеспечивается предсказуемое время выполнения системных функций: переключение и синхронизация задач, обработка прерываний и т. д. Это означает, что в коде, для которого критично время выполнения, исключаются алгоритмы с недетерминированным поведением, а все операции ожидания гарантированно ограничиваются тайм-аутами [3].

Помимо стандартного для ОСРВ функционала по управлению задачами и обеспечению их синхронизации, системные сервисы МАКС предоставляют разработчику возможности универсализированного ввода-вывода, поддержки командного терминала и др. Также в ОС предусмотрены встроенные средства отладки (профилировщик, доступ к системному времени, журналы событий). Используя готовые компоненты системы, разработчик может не тратить время на собственную реализацию необходимого вспомогательного функционала, а имеет возможность сосредоточиться на создании приложения.

Сегодня все чаще возникают ситуации, когда задачу лучше решать с помощью совокупности устройств, поэтому, в отличие от классических ОСРВ, в основу МАКС были положены принципы организации взаимодействия между несколькими устройствами (агентами), призванные упростить решение таких типичных задач, возникающих в распределенных системах, как распараллеливание вычислений, доступ к разделяемым данным, диагностика и резервирование устройств и т. д. Обычно для организации подобного взаимодействия используются стандартные примитивы Send и Receive, обеспечивающие отправку и прием сообщений (адресных или широковещательных), что позволяет создавать высокопроизводительные системы. Однако их реализация сопряжена с большими трудностями [4], особенно в ситуациях, когда необходим обмен сложными структурами данных.

Для упрощения решения уже давно была предложена концепция RPC (remote procedure call) — вызов процедур, расположенных на удаленных машинах. Базируясь на Send и Receive, RPC скрывает сложность, предлагая разработчику удобные в использовании абстракции. Однако есть и другие механизмы — альтернативная концепция

распределенной общей памяти [5], которая и была положена в основу ОСРВ МАКС.

Несколько независимых устройств могут обмениваться данными и синхронизировать их так, будто все они имеют физический доступ к общей памяти. Ключевым понятием здесь, в терминологии МАКС, является контекст — набор параметров, доступный из приложения на разных устройствах. Каждое из устройств может получать и обновлять данные контекста, не заботясь о наличии других

устройств, что обеспечивает ряд преимуществ. Например, когерентность данных в контексте может быть использована для повышения надежности системы: имея доступ к контексту, в котором сохраняется текущий статус задачи, резервное устройство продолжит выполнение задачи именно с того состояния, на котором неисправное устройство прекратило свою работу. В случае, когда допускается параллельное выполнение задачи группой устройств, использование разделяемых контекстов позволяет увеличить производительность. Кроме того, синхронизация контекстов решает проблему распределения подзадач внутри группы — каждое из устройств может резервировать для себя подзадачу в общем контексте и выполнять свою часть работы одновременно с другими. В общем контексте могут сохраняться также результаты работы и другие данные отдельных устройств, которые могут быть использованы всеми устройствами группы. В синхронизируемом контексте могут содержаться данные о функциональных возможностях доступных устройств и потребностях имеющихся задач. В этом случае у распределенной системы появляется возможность самостоятельно определять оптимальную конфигурацию — новое подключенное устройство автоматически узнает свою роль и выбирает посильную для себя задачу.

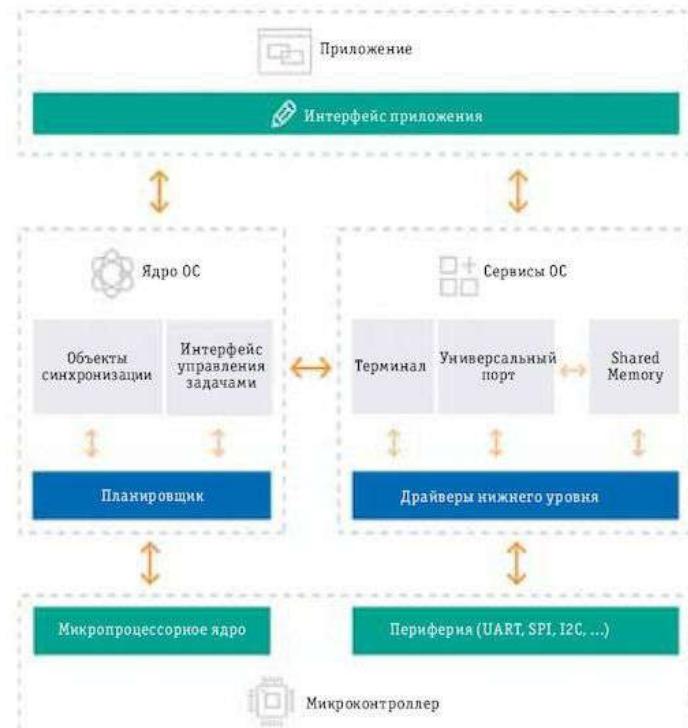


Рис. 1. Архитектура ОСРВ МАКС

Механизм работы с распределенной общей памятью позволяет группировке из нескольких устройств реализовать новый подход к выполнению задач. Представим, к примеру, группу роботов, убирающих большое промышленное помещение, — использование общего контекста позволит им совместно составлять карту здания, распределять рабочие зоны, обмениваться информацией о загрязненности различных участков и контролировать статус друг друга. В результате уборка будет выполняться быстрее и качественнее, чем при использовании нескольких независимых роботов, — минимизируется их простой и исключается появление неубранных участков. Оптимальное использование устройств достигается за счет автоматического перераспределения задач внутри группы в зависимости от их сложности (уровня загрязненности, топологии помещения и пр.) или при изменении состава устройств (подключение нового устройства и выход из строя существующего).

Реализация подобных механизмов возможна и без применения концепции распределенной памяти, однако в таком случае разработчику придется самостоятельно решать множество проблем, связанных с программированием с помощью стандартных механизмов обмена сообщениями. Концепция распределенной общей памяти позволяет

# в фокусе: операционные системы



Рис. 2. Контроль доступа и обработка ошибок в ОСРВ МАКС

упростить программирование — разработчик обеспечивает выполнение лишь несложных действий с памятью (запись, чтение), а ОС берет на себя обеспечение консистентности этих данных на всех узлах системы, организуя обмен данными и самостоятельно справляясь с возможными сбоями в узлах или каналах связи. Таким образом, ОСРВ МАКС позволяет реализовывать такие традиционно сложные механизмы, как резервирование управляющих устройств, совместные вычисления, обмен информацией с гарантией ее постоянной согласованности и др.

При разработке ОС особое внимание было уделено вопросам безопасности — ядро системы может задействовать доступные на целевом устройстве аппаратные средства защиты. Например, если в процессоре предусмотрена поддержка уровней привилегий при исполнении программ, то код ядра ОС всегда будет выполняться в привилегированном режиме, в то время как задачи приложения — в обычном. Это позволяет контролировать доступ приложения к аппаратным ресурсам (рис. 2). В зависимости от целевой платформы режим выполнения может ограничивать доступ приложения к некоторым инструкциям и регистрам процессора, системным модулям (управление прерываниями, таймер и т. д.), регистрам периферии или определенным областям памяти. Для повышения гибкости в планировщике предусмотрена возможность выполнения определенных задач с привилегиями ядра ОС, но только при указании разработчиком соответств-

ующего параметра задачи. Кроме того, при наличии в целевом процессоре блока защиты памяти система может задействовать его для ограничения доступа к определенным регионам памяти, а также для безопасного выявления ошибок приложения. Например, использование блока защиты памяти позволяет не только обнаружить переполнение стека, но и предотвратить при этом порчу данных.

К встроенным механизмам безопасности в ОСРВ МАКС можно также отнести и подсистему обработки ошибок в приложении. Программные и аппаратные исключения перехватываются ядром, а приложение оповещается о событии и имеет возможность корректно его обработать. После чего приложение должно выбрать соответствующее возникшей ситуации действие — завершить задачу, вызвавшую исключение, или попытаться продолжить ее исполнение. Это позволяет разработчику не только реализовать логику штатной работы устройства, но и описать альтернативные сценарии поведения приложения. При использовании подсистемы обработки ошибок совместно со встроенными инструментами отладки появляется возможность регистрировать все ошибки, возникающие в процессе работы приложения, и собирать сведения для отладки, что в итоге помогает быстрее локализовать дефект.

Одна из отличительных особенностей ОСРВ МАКС — объектно-ориентированный интерфейс, уже давно ставший нормой в разработке приложений для ПК, но еще не добравшийся до индустрии встраиваемых систем. В МАКС создана иерархия классов окружения системы, позволяющая структурировать приложения и упростить работу с системными сервисами. Для разработчиков, предпочитающих классический функциональный подход, в перспективе планируется обеспечить поддержку стандартных интерфейсов, не зависящих от конкретного оборудования CMSIS (Cortex Microcontroller Software Interface Standard) и подмножества POSIX

(Pthreads). Среди других направлений развития системы можно отметить расширение списка поддерживаемых платформ, включая ARM Cortex-M3/M4, ARM Cortex-M0/M0+/M1, TigerSHARC от Analog Devices, а также появление нового графического интерфейса. Кроме того, дальнейшее развитие получат механизмы организации взаимодействия между устройствами: поддержка сетей с ячеистой топологией (mesh) и реализация протоколов связи для Интернета вещей.

\*\*\*

В отечественной ОСРВ МАКС реализован необходимый для работы встраиваемых систем функционал, позволяющий не только ускорить разработку ПО для новых устройств на основе микроконтроллеров, но и организовать эффективное взаимодействие устройств в распределенных системах. Система МАКС проходит сертификацию в соответствующих ведомствах, что позволит гарантировать отсутствие в коде недекларированных возможностей. ■

## ЛИТЕРАТУРА

1. Алексей Федосеев. Системы реального времени: от Linux к Java // Открытые системы.СУБД. — 2009. — № 3. — С. 10–12. URL: <http://www.ospr.ru/os/2009/03/8112510> (дата обращения: 18.03.2017).
2. Леонид Акиншин, Анатолий Сысоев. Встраиваемые системы и патриотизм // Открытые системы.СУБД. — 2008. — № 8. — С. 68–71. URL: <http://www.ospr.ru/os/2008/08/5661718> (дата обращения: 18.03.2017).
3. Vivek Halwan, Jim Krodel. Study of Commercial Off-The-Shelf (COTS) Real-Time Operating Systems (RTOS) in Aviation Applications // Federal Aviation Administration. Research Reports. — 2002. — DOT/FAA/AR-02/118. — С. 2–3. URL: [https://www.faa.gov/aircraft/air\\_cert/design\\_approvals/air\\_software/media/AR-02-118\\_COTS.pdf](https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/media/AR-02-118_COTS.pdf) (дата обращения: 10.03.2017).
4. David R. Cheriton, Michael Stumm. The multi-satellite star: Structuring parallel computations for a workstation cluster. J. Distributed Comput. Memory Coherence in Shared Virtual Memory Systems | 359 (1988).
5. Kai Li, Paul Hudak. Memory coherence in shared virtual memory systems. ACM Transactions on Computer Systems (TOCS) 7.4. — 1989. — Р. 321–359.

Александр Кучеров ([alexander.kucherov@astrosoft.ru](mailto:alexander.kucherov@astrosoft.ru)) — руководитель проекта ОСРВ МАКС, «Астрософт» (Санкт-Петербург).

# ОС и СУБД: мандатное разграничение доступа



Применение ОС Linux для различных применений, требующих соблюдения конфиденциальности, делает актуальной задачу обеспечения безопасности связки ОС и СУБД, однако обеспечение сквозного мандатного разграничения доступа и независимости от платформы — весьма сложная задача.

*Ключевые слова: безопасность конфиденциальной информации  
Keywords: Astra Linux, PostgreSQL, Secure sensitive information, SELinux*

Валерий Попов

Разграничение прав доступа — важнейший элемент обеспечения безопасности информационной системы, однако сама по себе безопасность не самоцель, хотя об этом не всегда помнят. Если на одном компьютере разместить исключительно секретные материалы и физически отрезать его от сети, то проблемы с безопасностью будут решены. В ряде случаев так и поступают, однако выделять по компьютеру на каждую категорию секретности неразумно — иногда требуется, не покидая защищенную систему, иметь доступ ко всей информации, в том числе и несекретной. Каким образом, запуская в защищенной среде таких разных ОС, как Astra Linux Special Edition и SELinux/SEPGSQL, приложение, использующее СУБД PostgreSQL, обеспечить разграничение прав доступа и предоставить пользователю ровно тот уровень секретности, который ему положен? При этом очевидно, что ставить мандатную СУБД поверх немандатной ОС бессмысленно.

Всегда можно представить ситуацию, когда в большой базе данных лишь часть информации секретна и важно обеспечить гранулированность доступности. Например, сотрудники районных отделений полиции получают доступ к данным только о жителях своего района, а на уровне города должна быть доступна информация о любом жителе мегаполиса.

Такое разграничение доступа реализуется на уровне записей, или строк таблицы (Row Level Security, RLS), однако оно поддерживается не всеми СУБД.

В современных безопасных системах может быть реализована комбинация дисcretionного (избирательного) разграничения доступа (Discretionary Access Control, DAC), ролевого разграничения доступа (Role Based Access Control, RBAC) и мандатного (принудительного, обычно многоуровневого) разграничения доступа (Mandatory Access Control, MAC). Как правило, они реализуются именно в таком порядке: следующий «поверх» предыдущего. То есть ресурс, доступный по правилам мандатного доступа, заведомо доступен по правилам дисcretionного доступа, но не наоборот.

Мандатное управление доступом обсуждается редко, мало того, его иногда трактуют искаженно, опираясь на опыт работы с бумажными документами, для которых установлены различные уровни секретности. Перенос представлений о работе с бумажными документами на работу с электронными, на уровни доступа ОС и тем более на СУБД, иногда дезориентирует — сходство обманчиво. Для многоуровневой политики имеется простой принцип «читай вниз, пиши вверх», который не похож на то, что подразумевается в реальном мире. Принцип «Write up, read down. No read up, no write down» (субъект, обладающий определенным уровнем доступа, не может читать инфор-

мацию, относящуюся к более высокому уровню, но может читать менее секретные документы; субъект, обладающий определенным уровнем доступа, не сможет создавать объекты с более низким уровнем допуска, чем имеет сам, но при этом может писать в более высокоуровневые объекты), входящий в модель Белла — Лападулы, прописан и в отечественных документах, регламентирующих требования к безопасности информационных систем. Первая половина этого принципа очевидна, а про вторую этого сказать нельзя: невозможно представить себе ситуацию из «бумажного» мира, когда сотрудник получает доступ на запись в документ высокой секретности, не имея при этом права (и возможности) его читать. Однако именно так исключается попадание данных, доступных высокоуровневым объектам, в низкоуровневые объекты.

Мандатное разграничение доступа еще называют принудительным контролем доступа: пользователь не может управлять доступом к информации, а сами правила доступа жестко определены политикой безопасности. Наличие разграничения доступа MAC, DAC и RBAC — обязательное требование при защите информации категории «гостайна», однако для разных ОС и СУБД оно может трактоваться по-разному. Например, в мандатном управлении доступа для операционной системы Astra Linux пользователь сам может выбирать уровень секретности из тех, что разре-

# в фокусе: операционные системы

шены ему системным администратором, и этот уровень будет сохраняться на протяжении сессии. Теоретически мандатная система доступа не обязательно должна иметь различные уровни конфиденциальности (секретности или доступа), однако в реальной жизни редко находятся причины использовать немногоуровневую политику: именно она обязательна для систем с обеспечением гостайны.

Поток информации (документов) идет снизу вверх от менее секретных к более секретным — сотрудник, не имеющий формы допуска, может добавить свою часть отчета в общий, уже секретный отчет, не представляя себе общей картины. При этом сотрудник с формой допуска, позволяющей его прочитать, не может исправить ошибку в рабочих материалах первого сотрудника, ведь таким образом окажется раскрыта некая секретная информация, которой он располагает (повысится гриф первоначальной информации).

Имеется некоторый набор средств для реализации мандатных ОС, но для российской действительности актуальны две — SELinux [1] и Astra Linux Special Edition [2].

SELinux (Security-Enhanced Linux) представляет собой обычный дистрибутив Linux, скомпилированный с набором модулей (LSM, Linux Security Modules), перехватывающих системные вызовы. Модули представляют собой «крюки» («хуки»), к которым можно «подвесить» свои обработчики. Код SELinux открыт, и мандатный доступ строится поверх обычной системы доступа Linux — то есть файл, недоступный в Linux, будет недоступен и в SELinux, но не наоборот. При определении доступности файла система сравнивает метки субъекта и объекта, а затем принимает решение о допустимости операции. Метка представляет собой набор полей, содержимое которых может по-разному задаваться в различных политиках безопасности. Для создателей и пользователей мандатных систем наиболее интересна многоуровневая политика защиты (Multi Level Security, MLS), основанная на иерархии уровней секретности (чувствительности) и набора категорий. Пользователь SELinux — это сущность, определенная в политике, отвечающая за конкретный набор ролей и других атрибутов контекста безопасности. Пользователи SELinux отличаются от пользователей Linux, поэтому необходимо сопоставление (mapping) между ними через политику SELinux, позволяющее пользователям Linux наследовать ограничения пользователей SELinux.

Версия Astra Linux Special Edition разработана компанией «РусБИТех» на основе подсистемы PARSEC, целиком реализованной в виде модулей LSM, причем разработчики не декларируют следование модели Белла — Лападулы, а предлагают собственную патентованную «мандатную существенно-ролевую ДП-модель» (модель логического управления доступом «Д» и информационными потоками «П»). Объекты располагаются в монотонной по доступу иерархии контейнеров (каталог — это контейнер для файлов, таблица базы данных — контейнер для записей). Уровень конфиденциальности контейнера не уступает уровню содержащихся в нем объектов. По умолчанию запись «вверх» в модели безопасности Astra Linux невозможна — можно писать только на свой уровень. Однако, поскольку работать с такой жесткой системой запретов трудно, а в некоторых ситуациях просто невозможно, вводятся средства игнорирования мандатных меток контейнера или объекта. Атрибут «ehole» мандатной метки позволяет игнорировать любые мандатные правила, а бит CCR делает «прозрачными» контейнеры, непрозрачные для нижележащих уровней секретности.

## MAC В POSTGRESQL ДЛЯ SELINUX

В среде SELinux для поддержки разграничения доступа MAC для СУБД PostgreSQL имеется расширение sepgsql, которому необходима поддержка в ядре операционной системы, поэтому он может работать только в Linux 2.6.28 и выше. Это расширение работает на базе мандатной системы SELinux, используемой в Red Hat, CentOS, Fedora и др., а также в их российских собратах: ОС «Синергия-ОС» (РФЯЦ-ВНИИЭФ), ОС «Заря» и ряде других.

При принятии решения о предоставлении доступа к объекту базы данных, sepgsql сверяется с политиками безопасности SELinux, поскольку внутри sepgsql нет самостоятельного механизма назначения, хранения и модификации меток пользователей. Расширение обеспечивает присвоение меток безопасности пользователям и объектам базы данных через внешних «проводников», одним из которых является SELinux. Можно назначать метки безопасности схемам, таблицам, столбцам, последовательностям, представлениям и функциям. Когда расширение sepgsql активно, метки безопасности автоматически назначаются поддерживаемым объектам базы в момент их создания в соответствии с политикой безопасности, которая учиты-

вает метку создателя, метку, назначенную родительскому объекту создаваемого объекта, и, в некоторых случаях, имя создаваемого объекта. Для схем родительским объектом является текущая база данных; для таблиц, последовательностей, представлений и функций — схема, содержащая эти объекты; для столбцов — таблица.

Существующая реализация sepgsql имеет ряд особенностей и ограничений, которые необходимо принимать во внимание, но самое важное — это неспособность ограничения доступа на уровне строк. В версиях PostgreSQL 9.5 и 9.6 и во всех версиях Postgres Pro такая возможность предусмотрена.

## MAC В POSTGRESQL ДЛЯ ASTRA LINUX

Защищенные версии СУБД PostgreSQL, поставляемые компанией «РусБИТех» вместе с Astra Linux Special Edition v.1.5, собраны со специальными патчами, позволяющими взаимодействовать с мандатной системой PARSEC. Они базируются на стандартных версиях PostgreSQL 9.2 либо PostgreSQL 9.4 и отличаются реализацией мандатного разграничения (в 9.4 более полный функционал и реализована ДП-модель управления доступом и информационными потоками, соответствующая модели безопасности в ОС Astra Linux). СУБД PostgreSQL использует механизмы ОС для того, чтобы пользователь получил те же поля метки мандатного доступа, что и пользователь ОС, вошедший с соответствующими мандатными атрибутами. В сессии возможен только один уровень конфиденциальности, а не диапазон, как в SELinux и sepgsql.

В качестве главного контейнера выбрано табличное пространство pg\_global — одно на кластер базы данных. Применение мандатных прав доступа работает на уровне доступа к объектам базы данных и на уровне доступа непосредственно к данным. В отличие от sepgsql, эта реализация поддерживает разграничение на уровне записи: записи рассматриваются как объекты, а содержащие их таблицы — как контейнеры. Метки системных объектов располагаются в записях таблиц системного каталога, непосредственно описывающих защищаемый объект.

## «СИНЕРГИЯ-БД»

В реальности встречаются ситуации, когда sepgsql недостаточно, — например, когда требуется защита на уровне строк. В поставку ОС Astra Linux входит защищенная СУБД, привязанная к мандатной системе безопасности ОС Astra Linux. При этом

# в фокусе: операционные системы

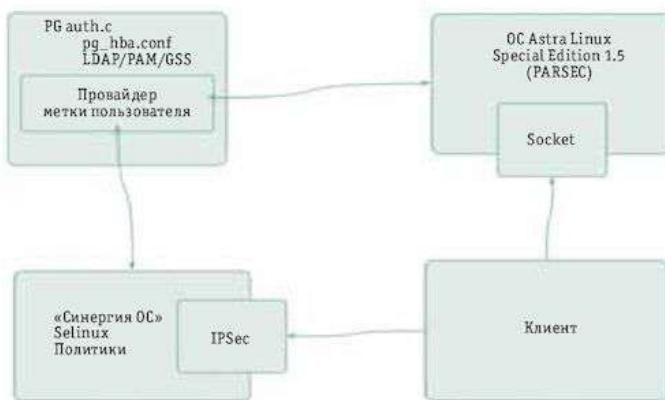


Рис. 1. Политики ОС управляют правами доступа удаленного клиента

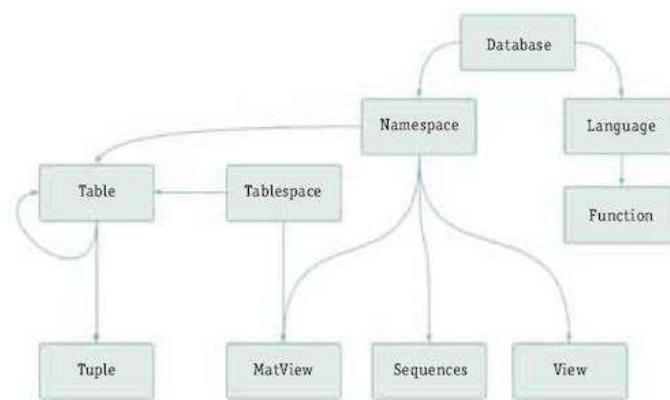


Рис. 2. Объекты-контейнеры и содержимое контейнеров в «Синергии-БД»

многие потребители нуждаются в защищенной, но платформенно-независимой СУБД. Кроме того, такую комбинацию СУБД и ОС проще сертифицировать: получить сертификат на комбинацию защищенной сертифицированной ОС и защищенной сертифицированной СУБД легче, чем на комплекс, в котором СУБД и ОС тесно взаимосвязаны. В последнем случае при изменении ОС (даже при переходе на другую версию той же ОС) придется заново начинать процесс инспекционного контроля сертифицированного продукта или процедуру сертификации.

Однако абсолютно кросс-платформенное решение невозможно — ОС, созданные даже на основе одного и того же ядра Linux, могут сильно отличаться [3]. Технически задача упростится, если исключить из набора платформ специфические ОС и взять за основу стандартные средства SELinux. Но тогда пришлось бы отказаться от поддержки платформы Astra Linux, популярной на ряде отечественных предприятий.

Возможно компромиссное решение — создать ПО связующего слоя, которое предоставит все необходимые для работы защищенной СУБД мандатные атрибуты, не требуя изменения кода ядра СУБД и исходного кода расширений ОС. Однако разработчикам придется потрудиться и над тем, чтобы универсальность не достигалась в ущерб производительности СУБД.

Мандатная СУБД «Синергия-БД», включающая в себя такое ПО промежуточного слоя, обеспечивает достаточную функциональность и приемлемую производительность (потери на уровне 15%). Эта СУБД разработана РФЯЦ-ВНИИЭФ и компаний Postgres Professional и представляет собой кросс-платформенную среду, работающую на отечественных системах «Синергия-ОС» и Astra Linux Special Edition.

Поскольку Astra Linux и «Синергия-ОС» по-разному решают проблемы дос-

тупа, то унификацию берет на себя ПО связующего слоя. Например, в Astra Linux пользователь регистрируется в системе с одним на сессию уровнем конфиденциальности, а «Синергия-ОС» предоставляет своим пользователям диапазон уровней — в этом случае «Синергия-БД» выбирает минимальный уровень из возможных. При этом, чтобы избежать деградации производительности СУБД, атрибуты безопасности пользователя кэшируются на время сессии. Пользоваться базой могут не только сотрудники с определенным уровнем доступа, заданным параметрами пользователя в ОС, но и те, кто входит в систему без метки доступа.

За основу «Синергии-БД» была взята версия СУБД PostgreSQL 9.5.5, в которой имеются штатные методы аутентификации, настраиваемые через pg\_hba.conf. На рис. 1 показан порядок взаимодействия удаленных пользователей с подсистемами безопасности двух разных ОС и «Синергии-БД».

Пользователь авторизуется через механизмы СУБД, например PAM (Pluggable Authentication Modules — «подключаемые модули аутентификации») или GSS (Generic Security Services — «общий программный интерфейс сервисов безопасности», например Kerberos), после чего запускается проверка механизма мандатного доступа. Например, если таблица-контейнер «прозрачна», то для любого пользователя она открыта для чтения и записи, но увидит он в ней только те строки, уровень конфиденциальности которых равен его собственному или меньше. Это и есть разделение доступа на уровне строк, отвечающее модели Белла — Лападулы.

На рис. 2 приведена иерархия объектов «Синергии-БД» — если наследуется таблица, то она считается логически входящей в контейнер родительской таблицы. Видно, что доступ к средствам процедурных языков базы данных (и тем более к функциям)

определяется на уровне базы данных, а не на глобальном уровне.

Модульная структура провайдеров безопасности дает возможность подключать новые модули и интегрировать СУБД в состав других защищенных ОС, имеющих мандатное разграничение доступа, что существенно упрощает и ускоряет процесс сертификации.

\*\*\*

Очевидно, что по разным причинам безопасные мандатные системы будут достаточно активно развиваться, а значит, будут развиваться подходы к обеспечению совместимости и кросс-платформенности ОС и СУБД. Кроме того, дополнительные исследования потребуются в области организации работы сетей в мандатной среде, обеспечения репликации, а также создания удобных интерфейсов взаимодействия с различными ОС и СУБД. ■

## ЛИТЕРАТУРА

1. Андрей Боровский. SELinux — система повышенной безопасности // Открытые системы. СУБД. — 2005. — № 4. — С. 30–37. URL: <http://www.osp.ru/os/2005/04/185543> (дата обращения: 18.03.2017).
2. Сергей Муравьев, Сергей Дворянкин, Игорь Насенков. СУБД: проблема выбора // Открытые системы. СУБД. — 2015. — № 1. — С. 22–24. URL: <https://www.osp.ru/os/2015/01/13045322> (дата обращения: 10.03.2017).
3. Алексей Гриневич, Денис Марковцев, Владимир Рубанов. Проблемы совместимости Linux-систем // Открытые системы. СУБД. — 2007. — № 1. — С. 10–15. URL: <https://www.osp.ru/os/2007/01/3999198> (дата обращения: 18.03.2017).

Валерий Попов ([v.popov@postgrespro.ru](mailto:v.popov@postgrespro.ru)) — руководитель группы информационной безопасности и сертификации, компания Postgres Professional (Москва).

# Риски пользовательских интерфейсов

Манипулирование — одно из самых вопиющих нарушений свободы личности, неизменно унижающее достоинство человека. Добывая идентифицирующие сведения, интеллектуальные пользовательские интерфейсы создают угрозу неприкосновенности личности и другие риски.

Ключевые слова: интеллектуальные интерфейсы, персональные данные, социальная инженерия  
Keywords: intelligent user interface, IUI, personal information, social engineering

Кристофер Хазард,  
Муниндар Сингх

**И**нтеллектуальный пользовательский интерфейс (*intelligent user interface, IUI*) — это обычно приложение (или его часть), в процессе взаимодействия с пользователем реагирующее на изменения его потребностей. Иными словами, IUI выполняет свои функции, оперативно адаптируясь к действиям конкретного пользователя. Такие интерфейсы работают, конструируя и непрерывно обновляя модель пользователя и контекста, а также регистрируя определенные аспекты пользовательского профиля (например, демографические данные), историю его взаимодействий и коммуникаций, цели, предпочтения, социальные связи, черты характера, а также показатели физиологического и психологического состояния.

Примерами IUI могут служить календари и навигационные приложения (Google Now и т. п.), цифровые помощники (Apple Siri), игры для стационарных и мобильных устройств (Pokemon Go). Ясно, что IUI могут эффективно функционировать только благодаря информации, собираемой о пользователе, которая может предоставляться напрямую им самим (пол, возраст и т. д.), извлекаться с его согласия из других сервисов (содержание электронной почты, списки друзей и т. п.), регистрироваться в процессе взаимодействий с пользователем (поисковые запросы и их результаты). Данные могут также собираться косвенным путем (сведения о местах, которые пользователь посещал, например, в процессе

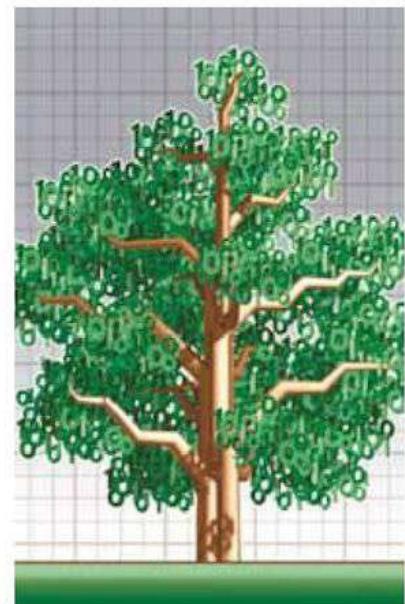
игры), сохраняться в результате прямых обращений (вроде вопросов о том, хотел ли бы пользователь принять конкретный звонок), а также формироваться путем анализа взаимодействий (из которого может быть сделан вывод о том, что утром и поздно ночью пользователь предпочитает менее интерактивный контент). Обладая всей этой информацией, IUI старается улучшить обслуживание — интерфейс выясняет цели пользователя, его предпочтения и вносит соответствующие изменения.

Делая предположения о поведении конкретного пользователя либо запоминая закономерности действий всей аудитории, IUI может получать дополнительные сведения. Например, легко предположить, что дом или работа — это места, где пользователь чаще всего бывает или именно оттуда выясняет маршруты до других мест [1]. Кроме того, пользователи обычно ежедневно передвигаются по одним и тем же маршрутам, то есть траектории перемещений позволяют однозначно идентифицировать человека.

По мере осознания проблемы соблюдения приватности соответствующие риски привлекают внимание как общественности, так и властей: с увеличением объемов и разнообразия данных, собираемых IUI о пользователях, растут угрозы раскрытия персональной информации, вторжения в личное пространство и потери репутации [2].

## ПОТЕНЦИАЛЬНЫЕ ПРЕИМУЩЕСТВА IUI

Чтобы экономить время пользователя, ему нужно обеспечить более мощную под-



держку в принятии решений, а для этого приходится учитывать потребности конкретного индивидуума, для чего и строится его детализированная модель. Проще говоря, IUI из многочисленных вероятных вариантов действий выбирает те, которые в наибольшей мере соответствуют целям пользователя. Например, если отправитель письма, находящийся в Роли (шт. Северная Каролина), ищет адрес в Дареме, то, скорее всего, навигационное приложение автоматически выберет город Дарем в том же штате, а не одноименный город в Великобритании. В 2013 году СМИ написали о жительнице Бельгии, которая из-за ошибки GPS-навигатора пересекла всю Европу, вместо того чтобы проехать всего 140 км, — возможно, IUI помог бы этого избежать. Однако реализация IUI требует интеллектуальных функций, а не фиксированных правил. Например, если в упоминавшемся письме говорится о бронировании билета до аэропорта Дарем Долина Тиса, то тогда речь все-таки идет о британском Дареме.

Игровое или учебное приложение, пользуясь аналитикой, может выбирать задачи для пользователя в зависимости от уровня его усталости или компетенции: более опытные игроки или студенты получают более сложные задания, чтобы не скучать, а остальные — попроще, чтобы их не отпугнула чрезмерная сложность. Это не что иное, как применение психологической концепции «потока» (периода максимальной продуктивности, достигаемой при определенных условиях), которая широко используется при создании игр.

## РИСКИ ПРИВАТНОСТИ

Вкратце можно сказать, что все сложности, связанные с IUI, сводятся к тому, что для эффективной работы интерфейсу нужно получить или логически вывести достаточно большой объем информации о пользователе. И наибольшую ценность представляют именно сведения личного характера, компрометация которых способна создавать угрозу для безопасности пользователя, его финансов или репутации (скажем, студент с самой низкой успеваемостью вряд ли захочет, чтобы все в группе об этом узнали).

Нарушения приватности могут произойти вследствие самых разных причин. Например, если вы сохранили местонахождение своего дома в навигационном приложении на смартфоне, то вор, укравший его, сможет найти эту информацию и ограбить вас. Но есть еще риск атак, совершаемых через сами приложения: например, навигационное приложение специально прокладывает маршрут через бар, справедливо предполагая, что, когда вы устанете после долгой прогулки или поездки, этот намек заставит вас зайти именно в данное заведение.

## Извлечение и распространение информации

Извлекать информацию из обученных моделей можно даже в том случае, когда недоступны исходные данные. Модели машинного обучения — это по сути скатая презентация части данных о пользователе потенциально личного или конфиденциального характера. Во многих случаях пользователь даже не подозревает о факте сбора личных данных о себе, поскольку они скрыты за обычными. Допустим, к примеру, что IUI изучает предпочтения пользователя в целях повышения продуктивности его работы. При этом интерфейс может выяснить о пользователе что-то, не относящееся напрямую к задаче, — например, время, когда тот просыпается утром, или период в течение дня, когда он непродуктивен. При этом ни сам пользователь, ни даже разработчик приложения, возможно, не знают о том, что в собираемых данных содержатся такие сведения. Но если бы IUI раскрывал их для других, то, например, из вашего календаря вашему начальнику стало бы известно, что вы начали работать не в 8 утра, а в 10. Неменьший ущерб был бы нанесен, если бы календарь оповестил ваших клиентов, что в 8 часов вы все же доступны, но не желаете с ними пообщаться, так как выделяете это время для более важных дел.

## Зондирование пользователей

Для IUI недостаточно пассивного наблюдения за пользователем — система может «прощупывать» его, предлагая тщательно подобранные альтернативы, чтобы выяснить его физиологическое или психологическое состояние. В зависимости от сделанного им выбора, IUI может узнать, находится ли пользователь в подавленном состоянии, сидит ли на диете, а также, предлагая достаточно много вариантов выбора, получить другие сведения, используя усталость пользователя от принятия решений. Некоторые исследования ставят под сомнение существование этого явления, которое психологи также называют истощением, однако приложение может быстро накопить базу эмпирических результатов, гораздо более обширную, чем собирается в рамках локальных научных исследований, и, возможно, разработчик захочет извлечь коммерческую выгоду из полученных данных. Если их анализ покажет, что с помощью IUI можно как-то воздействовать усталость от принятия решений, есть вероятность того, что пользователем будут манипулировать.

## Компрометация безопасности и личности

Многие протоколы аутентификации полагаются на сообщение секретов, известных обеим сторонам. Например, для проведения транзакции по банковской карте покупатель для подтверждения личности должен сообщить свой домашний адрес. А если что-то не совпадает, эмитент может попросить пользователя напомнить, какие платежи он совершил в различных местах, предполагая, что эти сведения могут быть известны только реальному держателю карты. Но приложение, следящее за местонахождением, может определить ваш домашний адрес, а также магазины, где вы были и, возможно, делали покупки.

## Прямое манипулирование

Прямое манипулирование — это попытки получения частичного или полного контроля над действиями пользователя, характеризующиеся средней или высокой вероятностью успеха для каждого конкретного случая. Иначе говоря, если пользователем манипулируют подобным образом, есть высокая вероятность убедить его делать что-то, ему не свойственное. Для этого приватная информация нужна не всегда, но доступность частных или устанавливающих личность данных повышает шансы на успех. Возможен и вариант, когда сама манипуляция позволяет получить такие сведения.

К числу прямых атак против свободы личности относятся манипулятивные приемы оформления интерфейса (dark pattern), наводящие пользователя на желательное для разработчиков действие. Пример — навигационное приложение, которое многократно, пока вы не ответите утвердительно, спрашивает, разрешаете ли вы сервис-провайдеру регулярно собирать подробные данные с вашего смартфона в целях улучшения результативности. Досаждая пользователю, такое приложение по сути склоняет его дать согласие; впоследствии тот может забыть, что дал разрешение, или не найти способа отозвать его.

## Косвенное манипулирование

Косвенное манипулирование — это попытки получить частичный контроль над действиями пользователя, характеризующиеся крайне малой вероятностью успеха в каждом конкретном случае. Другими словами, для успеха требуются либо воздействие на очень большую аудиторию или многократные воздействия на одного и того же пользователя, либо то и другое.

Примеры косвенного манипулирования — рекламные объявления, а также особенности оформления интерфейсов, оборудования и т. п., результатом чего становится незначительные изменения поведения. При разработке интерфейса предпочтительный для разработчика выбор выделяется или делается более простым. Например, во многих казуальных играх есть встроенные покупки, которые позволяют быстрее продвигаться вперед на сложных или скучных этапах. Разработчик может предложить игроку возможность купить предмет, увеличивающий шансы прохождения трудного этапа, в самый подходящий для этого момент. Собирая общие данные об аудитории игроков, для извлечения дополнительной выгоды можно с помощью аналитики выяснить, когда вероятность совершения покупки выше и как удержать играющего в игре, если он решил ее прекратить.

Агрегированные данные о пользователях помогают в косвенном манипулировании, предоставляя разработчикам средства оценки, классификации и сегментирования целевой аудитории, а также эмпирической проверки результатов опосредованных воздействий.

Казуальные мобильные игры — типичный пример IUI, способного понизить самоконтроль пользователя, увеличивая его когнитивную нагрузку, и, пользуясь этим состоянием, подталкивать его к принятию машинных действий. Так, популярные

# безопасность

игры вроде Clash of Clans, Candy Crush Saga и Pokemon Go вынуждают принимать массу решений, в отдельности представляющихся важными, но в целом не влияющих на общий ход игры. Вообще, создатели игр обычно стараются извлекать выгоду за счет улучшения впечатлений играющего, но недобросовестный разработчик может эксплуатировать реакции пользователя, предлагая принимать решения в моменты, когда тот находится в невыгодном положении.

## СНИЖЕНИЕ РИСКОВ

Как можно ограничить описанные риски, не теряя преимуществ IUI?

## Этичный дизайн пользовательского интерфейса

Отраслевые стандарты, социальные нормы, законодательные акты, оптимальные методы и сертификации в совокупности позволили бы ограничить риски нарушения приватности при создании коммерческих сервисов.

## Архитектурные решения и открытые стандарты

Хорошо проработанная архитектура и алгоритмы позволяют защитить приватность, в то же время предоставляя сервис-провайдерам доступ к необходимым для IUI данным и аналитике. Методы дифференциальной приватности (анализ сведений о большом числе людей без идентификации личности) при определенных условиях обеспечивают защиту за счет добавления «шума» или изменения дискретизации данных. Существует также связующее ПО, предоставляющее приложениям с IUI высокоразвитый API, маскирующий идентифицирующую информацию о пользователях и уведомляющий лишь о готовности пользователя к действию приложения, выполняемому с учетом результатов анализа. Два этих подхода можно адаптировать для IUI, ослабив связь между решениями, которые нужно принимать в игре, и состоянием пользователя.

## Пользовательские агенты

Специальный код для защиты интересов пользователя, действующий на доверенной платформе, поможет защититься от угроз приватности, исходящих от IUI. Похожие методы оказались действенными, в частности, против распознавания пользователей путем «дактилоскопии» браузера — например, есть плагины, которые случайным образом меняют идентифицирующие заголовки. Для IUI понадобятся агенты, рабо-

тающие с более сложными угрозами, чем просто возможность отследить действия пользователя.

Агенты могли бы фильтровать данные, предоставляемые интерфейсу, или предупреждать пользователя при возникновении риска компрометации личных сведений. Скажем, определять, какие поля ввода данных необходимы сервису, а какие с учетом пользовательских интересов заполнять рискованно.

Агенты могли бы также фильтровать данные для систем обработки информации, просматривая переданный контент и вызовы API — например, следя за привилегиями, которые запрашиваются мобильными приложениями. Или выполнять роль межсетевого экрана с распознаванием контента, анализирующую и фильтрующую данные перед отправкой сервис-провайдеру. Скажем, если игра передает список контактов пользователя третьей стороне, такой агент мог бы заблокировать отправку.

Для агентов понадобятся открытые архитектуры, но идея может оказаться нереализуемой, так как операторы платформ и поставщики контента, пользуясь юридическими и техническими барьерами, ограничивают пользователя в возможности взаимодействия с программным обеспечением с помощью автоматизированных средств. Возможно, необходимую открытость можно было бы обеспечить путем государственного регулирования.

## Экономические модели

Захист приватности в среде с враждебными элементами и ресурсными конфликтами может потребовать значительных затрат. Личная информация о человеке представляет ценность во многих контекстах, а IUI станут одним из главных полей «битвы» между обеспечением приватности и стремлением к раскрытию сведений, а также между противоборствующими сторонами, такими как сервис-провайдеры и блокировщики рекламы.

Методы теории игр, учитывающие стратегии конкурирующих игроков (в данном случае поставщиков IUI и пользователей), могут помочь в разработке механизмов, обеспечивающих оптимизацию решения ряда задач. Для IUI также можно было бы адаптировать некоторые защитные методы, разработанные для физической инфраструктуры.

## Контроль происхождения и аудит

Если сохранять сведения о том, какие данные послужили основанием для тех

или иных выводов и действий, можно будет проверять, была ли информация задействована без согласия пользователя. Технологии блокчейна дают возможность хранить данные так, чтобы производить вычисления над ними и удостоверяться в их надежности могли только держатели соответствующих ключей шифрования. Реализуемый на основе блокчейна механизм «умных» контрактов можно доработать, обеспечив возможность без раскрытия содержания выяснить, на основании каких данных совершилась та или иная транзакция либо делался аналитический вывод. Допустим, к примеру, что поставщик IUI по условиям контракта обязан объяснять все свои решения. То есть интерфейс может получать личные данные пользователей, но аналитические выводы и решения нужно сохранять в блокчейне со ссылками на конкретные данные, ставшие их основанием. Пользователь, выполнив с помощью специальных средств аудит блокчейна, может выяснить, применялись ли его данные для целей, не оговоренных в контракте. Но этот метод далек от идеала: по связям между личными данными, хранимыми в блокчейне, можно узнать конфиденциальную информацию о пользователе и коммерческие секреты поставщика IUI.

\*\*\*

Манипулирование — одно из самых вопиющих нарушений свободы личности: достаточно вспомнить хотя бы, каких масштабов достиг скандал, когда в Facebook провели психологический эксперимент над почти 700 тыс. пользователей, специально меняя состав ленты новостей, чтобы выяснить, как это влияет на их эмоциональное состояние. Добывая идентифицирующие сведения, IUI создает угрозу неприкосновенности личности и другие, не менее коварные риски, понимание и устранение которых станет ключом к дальнейшему развитию интеллектуальных интерфейсов. ■

## ЛИТЕРАТУРА

1. R. Liu et al. An Unsupervised Collaborative Approach to Identifying Home and Work Locations. Proc. 17th IEEE Int'l Conf. Mobile Data Management, 2016. doi:10.1109/MDM.2016.53.
2. W. L. Prosser. Privacy // California Law Rev. — 1960. Vol. 48, № 3. — P. 383–423.

Кристофер Хазард ([cjhazard@hazardoussoftware.com](mailto:cjhazard@hazardoussoftware.com)) — основатель компании Hazardous Software; Муниндар Сингх ([singh@ncsu.edu](mailto:singh@ncsu.edu)) — профессор, Университет Северной Каролины.

# Создание критически важных приложений на основе микросервисов

Ошибки в системном ПО непременно будут использоваться для атак, поэтому критические приложения не должны зависеть от корректности ПО низкого уровня. Применение микросервисов и защищенных областей памяти, таких как Intel Software Guard Extension, минимизирует доверенную вычислительную базу и обеспечивает требуемую надежность приложений без ущерба производительности.

*Ключевые слова:* контейнерная виртуализация, надежность и отказоустойчивость, микросервисы  
*Keywords:* Container Virtualization, Intel Software Guard Extensions, Microservices, Reliability and fault tolerance

Кристоф Фетцер

**С**тандартный подход к созданию ответственных приложений для бизнеса и критически важной инфраструктуры — это подход «снизу вверх», когда начинают с надежного фундамента, выбирая аппаратную архитектуру и системное ПО, а поверх него разрабатывают приложение и обеспечивают его безопасное выполнение. Убедиться в надежности микропрограммного и системного ПО (ОС, гипервизор, диспетчер ресурсов и т. д.) помогают формальные методы доказательства корректности микроядерной системы [1].

Под управлением Linux, операционной системы с монолитным ядром, сегодня работают многие современные системы и устройства: облачные инфраструктуры, смартфоны, автомобильные компьютеры и другие устройства, критичные к безопасности встроенного программного обеспечения [2]. Однако доказать кор-

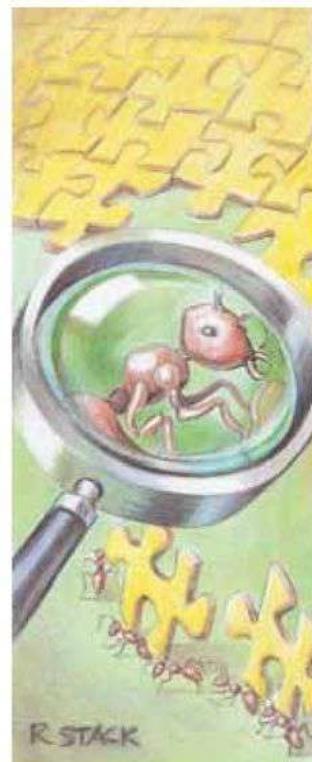
ректность работы ОС Linux современными формальными методами практически нереально — если в самом микроядре насчитывается лишь 10 тыс. строк кода, то для ядра Linux пришлось бы верифицировать более 20 млн строк. Более того, опыт показал, что Linux всегда содержит исправимые ошибки, так что формальное доказательство корректности в любом случае даст отрицательный результат.

Статический анализ репозиториев открытого кода обычно выявляет в среднем 0,61 дефекта на тысячу строк, и, несмотря на постоянно проводимые исправления, число неустранимых ошибок в ядре остается на уровне примерно 5 тыс. [3]; правда, не все они могут использоваться для проведения атак. Другое исследование показало, что за последние пять лет в Linux было исправлено около 500 ошибок, связанных с нарушением безопасности, и они присутствовали в ядре в течение пяти лет [4]. В проприетарном коде плотность дефектов несколько выше,

чем в открытых проектах, а значит, коммерческое ПО тоже не застраховано от наличия уязвимостей.

## ЗАЩИТА И БЕЗОПАСНОСТЬ ПРИЛОЖЕНИЙ

Ошибки в низкоуровневом системном ПО могут использоваться для атак, поэтому ответственные приложения нужно создавать так, чтобы их конфиденциальность и целостность не зависели от корректности системного ПО. В случае важных бизнес-приложений или приложений, критичных к безопасности, нужны также гарантии корректности работы центральных процессоров, сложность которых в последние десятилетия резко возросла. Современные процессоры имеют миллиарды транзисторов и подвержены не только спорадическим ошибкам, но и дефектам, привнесенным на стадии проектирования. Жесткий параллелизм (когда два ядра параллельно выполняют одни и те же команды для взаимной кор-



# программная инженерия

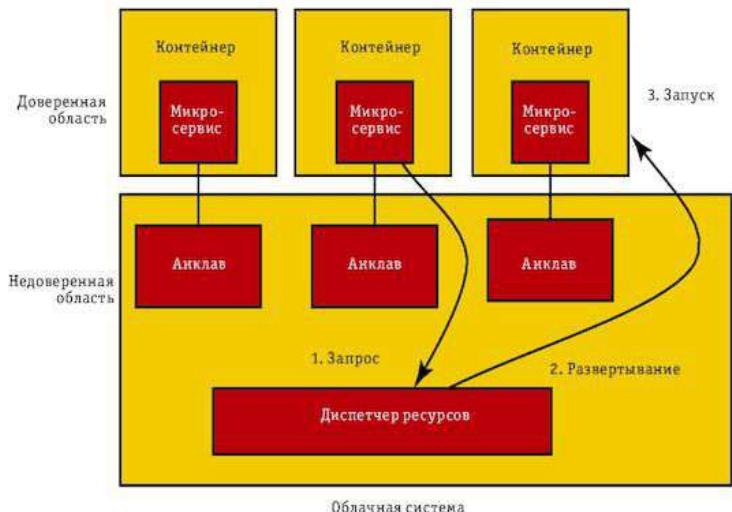


Рис. 1. Обеспечение компактности состояния, хранимого в анклаве, с использованием микросервисов. Такой подход позволяет свести к минимуму доверенную вычислительную базу и поддерживать достаточное быстродействие

рекции ошибок), применяемый в мэйнфреймах и встроенных системах, позволяет распознавать нерегулярные ошибки, но не конструктивные дефекты. Процессоры старшего класса отличаются высокой степенью недетерминированности, из-за чего практически невозможно обеспечить жесткий параллелизм без существенного снижения быстродействия и серьезных изменений в конструкции.

Если при создании драйверов разработчики полагаются на надежность операционной системы, то при написании критически важных приложений они доверяют только подконтрольным им компонентам. При выполнении ответственного приложения в облаке системные сервисы и ОС могут находиться под административным контролем третьей стороны. Современные расширения набора инструкций процессора, в частности Intel Software Guard Extensions (SGX), позволяют приложениям сохранять состояния в защищенных областях памяти — «анклавах», доступа к которым нет даже у самого привилегированного кода, включая ОС и гипервизор. Однако SGX более чем на порядок замедляет выполнение своих команд, если рабочий срез данных анклава не помещается в расширенный кэш страниц SGX, размер которого составляет сейчас лишь около 90 Мбайт. Поэтому данные состояния для анклава нужно ограничивать в объеме — не только для минимизации базы доверенных вычислений (совокупности элементов, влияющих на защищенность системы), но и для того, чтобы сохранить приемлемую производительность. Для выполне-

ния этих требований была реализована и проверена архитектура на основе микросервисов (рис. 1).

## ПРИЛОЖЕНИЯ НА ОСНОВЕ МИКРОСЕРВИСОВ

Архитектура микроядра, пожалуй, более всего подходит для создания систем повышенной надежности. Тем не менее такие системы нередко создаются на базе монолитных ядер наподобие Linux, поэтому при создании надежных систем следует опираться на использование унаследованных программных и аппаратных компонентов, которым можно доверять до известной степени, — так проще обеспечить целостность, конфиденциальность и корректное выполнение кода.

В этом случае можно разбить приложения на серию микросервисов, отвечающих за определенную функцию. Сами микросервисы слабо связаны и поддерживают горизонтальное эластичное масштабирование: сбои и деградация быстродействия устраняются путем запуска новых экземпляров сервиса вместо аварийных или замедлившихся. Микросервисы хорошо сочетаются с современными фреймворками управления контейнерами, такими как Kubernetes и Docker Swarm, имеющими средства

обеспечения высокой доступности, балансировки нагрузки и масштабирования приложений. Пока еще нет версий для встроенных систем и автомобильных компьютеров, но в ближайшее время они должны появиться.

Разберем теперь, как обеспечить целостность, конфиденциальность и корректность выполнения микросервисов при помощи программных средств и расширений команд процессора.

## ОБЕСПЕЧЕНИЕ КОРРЕКТНОСТИ

Если приложение требует достаточно высокой надежности (скажем, допускается максимум один системный сбой за  $10^9$  часов эксплуатации), то на корректную работу центрального процессора уже рассчитывать нельзя. Несмотря на то что современные процессоры распознают и маскируют большинство нерегулярных ошибок, для критически важных приложений их уровень распознавания еще недостаточен. Поскольку высокопроизводительные процессоры отличаются большой степенью недетерминированности, традиционный жесткий параллелизм здесь нереализуем — нужны программные средства синхронизации. Преимущество такой программной защиты в том, что ее можно комбинировать с механизмом быстрого восстановления на основе транзакционной памяти процессоров. Как показал опыт, этот способ позволяет распознавать и маскировать большинство нерегулярных ошибок. В данном случае накладные затраты ресурсов примерно такие же, как

при аппаратной синхронизации, но при этом маскируется около 90% нерегулярных ошибок и поддерживаются недетерминированные процессоры и приложения [5].

Для некоторых критически важных приложений необходимо распознавать еще и конструктивные дефекты процессора, используя

### Микросервисы: простые серверы, сложная система безопасности

Преимущества архитектуры микросервисов — небольшие команды разработчиков, ускорение циклов выпуска обновлений, сокращение зависимостей и уменьшение рисков. Однако вопросы безопасности, с которыми приходится сталкиваться при использовании новой парадигмы, менее проработаны.

Сердар Егудалл  
Computerworld Россия, 2016, № 06

программную синхронизацию (каждая команда выполняется дважды) в сочетании с арифметическим кодированием. Данный способ позволяет с высокой надежностью распознавать ошибки процессора, а накладные расходы при этом не превышают 150%. Примерно такой

же результат дает аппаратно реализованный жесткий параллелизм с учетом того, что синхронизируемые ядра приходится замедлять, чтобы синхронизация сохранялась после коррекции ошибки одного из ядер.

## ЦЕЛОСТНОСТЬ И КОНФИДЕНЦИАЛЬНОСТЬ

Для обеспечения конфиденциальности и целостности микросервисов требуется защищенный контейнер. С точки зрения контейнерного движка (в данном случае имеется в виду Docker без дополнений) защищенный контейнер не отличается от обычного. В каждом защищенном контейнере в анклавах SGX выполняется один экземпляр микросервиса (рис. 2). Регистры процессора, основная память, файлы и сетевые коммуникации прозрачно шифруются, благодаря чему обеспечиваются конфиденциальность и целостность, в том числе защита от атак со стороны более привилегированного ПО, например гипервизора и ОС. При этом быстродействие близко к скорости работы системы без виртуализации при условии, что анклавы умещаются в ЕРС.

Микросервисы упрощают защиту от атак, совершаемых через интерфейс системных вызовов (например, атак уровня ядра, основанных на манипуляциях с результатами вызовов). Для улучшения масштабируемости и снижения числа зависимостей микросервисы могут частично замещать сервисы ОС (например, сервис файловой системы). Многие микросервисы пользуются лишь ограниченным подмножеством системных вызовов, которые можно прозрачно защитить с помощью контейнерной инфраструктуры.

Приложения можно создавать на основе уже имеющихся микросервисов или путем разработки новых. Например, в приложении можно задействовать сервисы memcached или Redis. Многие существующие сервисы написаны на небезопасных языках, к каковым можно отнести Си и C++. Поэтому, как и ядро ОС, такие сервисы нужно защищать от совершаемых через сетевые API атак, таких как переполнение буфера или использование форматирующих строк. В решении задачи обеспечения безопасности поможет механизм для компиляторов, реализующий дополнительную защиту существующего кода. Поскольку критически важные приложения должны быть высокодоступными, созданный механизм терпим к сбоям, например к ошибкам доступа, но останавливает сервис только в том случае, если

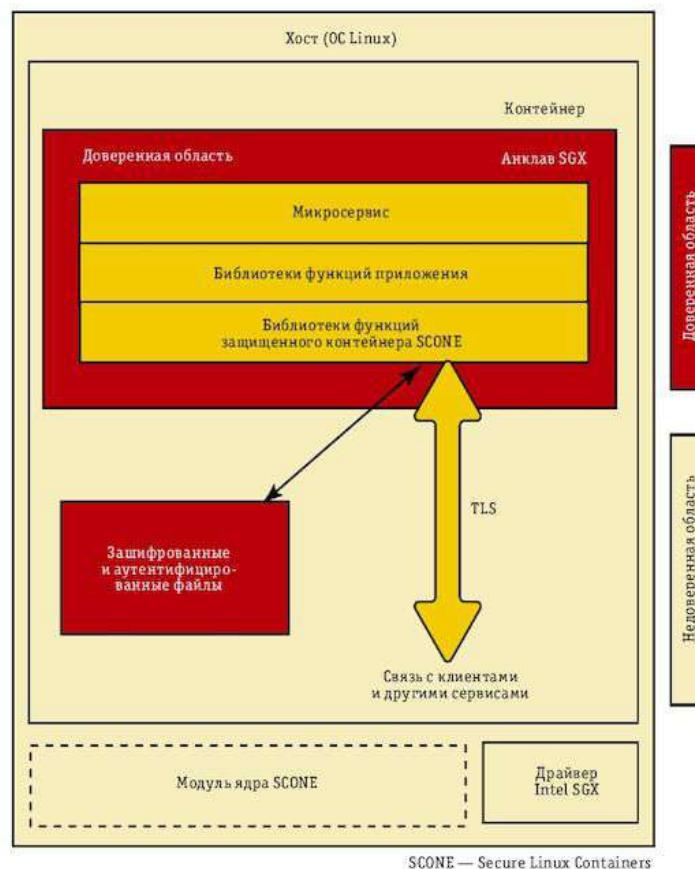


Рис. 2. Каждый защищенный контейнер выполняется внутри отдельного экземпляра микросервиса в анклаве SGX, благодаря чему соблюдаются конфиденциальность и целостность и исключаются атаки со стороны программного обеспечения с более высокими привилегиями (ОС или гипервизора)

продолжение его работы создает угрозу безопасности.

\*\*\*

Применение микросервисов в сочетании с защищенными контейнерами открывает новый путь к созданию критически важных приложений, позволяющий сохранить возможность использования инструментов и сервисов, разработанных для ПО, менее требовательного к надежности. При этом характеристики, необходимые высоконадежным приложениям, обеспечиваются с помощью защищенных контейнеров и специальных расширений компилятора. Предлагаемый подход может использоваться для реализации приложений, автоматически прекращающих работу при сбое, однако его можно применять и для создания ПО, сохраняющего работоспособность при единичных отказах. ■

## ЛИТЕРАТУРА

1. G. Klein, G. Heiser, T. Murray. It's Time for Trustworthy Systems. IEEE Security & Privacy. — 2012. — Vol. 10, № 2. — P. 67–70.

**2.** L. Bulwahn, T. Ochs, D. Wagner. Research on an Open-Source Software Platform for Autonomous Driving Systems, tech. report. BMW Car IT GmbH. Oct. 2013. URL: <http://www.bmw-carit.de/downloads/publications/ResearchOnAnOpenSourceSoftwarePlatformForAutonomousDrivingSystems.pdf>. (дата обращения: 01.03.2017).

**3.** Coverity Scan Open Source Report 2014, Coverity Scan. 2015. URL: <http://go.coverity.com/rs/157-LQW-289/images/2014-Coverity-Scan-Report.pdf> (дата обращения: 01.03.2017).

**4.** K. Cook. The State of Kernel Self Protection Project. Linux.com. — 12 Sept. 2016. URL: [www.linux.com/videos/state-kernelself-protection-project-kees-cook-google](http://www.linux.com/videos/state-kernelself-protection-project-kees-cook-google) (дата обращения: 01.03.17).

**5.** D. Kuvaikii et al. HAFT: Hardware-Assisted Fault Tolerance. Proc. 11th European Conf. Computer Systems (Eurosys 16). — 2016. — Article 25.

Кристоф Фетцер  
(chrostof.fetzer@tu-dresden.de) —  
Дрезденский технический университет (ФРГ).

# Интернет боевых вещей

С расширением Интернета вещей входящим в него гражданским и военным системам потребуются почти беспредельные способности масштабирования.

*Ключевые слова: Интернет боевых вещей, Интернет вещей, Интернет всего, Интернет людей  
Keywords: Internet of Battle Things, Internet of Everything, Internet of People, Internet of Things*

Александр Котт,  
Анантрам Свами, Брюс Вест

**С**тремительному распространению Интернета вещей (Internet of Things, IoT) способствует развитие машинного интеллекта и сетевых коммуникаций, причем «вещи» приносят больше пользы, когда они активно обмениваются информацией друг с другом. Это касается и интеллектуальной техники на полях боевых сражений — Интернета боевых вещей (Internet of Battle Things, IoBT): обмениваясь данными, такие «вещи» могут быть полезны солдатам в бою. В некоторых отношениях IoBT уже реальность [1].

На полях сражений будущего будут действовать всевозможные устройства, как «разумные», так и не очень, которым предстоит решать широкий круг задач, регистрируя информацию и взаимодействуя друг с другом и людьми [2]. Среди этих устройств будут датчики, снаряжение, оружие, транспортные средства, роботы и носимая техника (рис. 1), способные избирательно получать и обрабатывать информацию, выполнять посреднические функции при выяснении сути данных, вести скординированные оборонительные операции, а также различными способами воздействовать на противника. Все эти задачи будут решаться совместно — устройства станут непрерывно общаться, координировать и согласовывать свои действия, разрабатывая и выполняя задания.

Чтобы эта грандиозная картина стала реальностью, требуется решить целый ряд задач — в частности, обеспечить между вещами гибкую связь, которая бы адаптировалась к условиям быстро меняющихся ситуаций на поле боя. Для этого понадобится организовать управ-

ление большим количеством динамичных активов (устройств, каналов и т. п.), допуская при этом множество сложных компромиссов. Адаптация сети, управление ею и ее реорганизация должны проходить по большей части автономно, без привлечения людей для ее поддержки и сопровождения.

Кроме того, необходимость разбираться в потоках информации, генерируемой IoBT, сильно усложнила бы выполнение боевой задачи для людей, находящихся в условиях экстремальной когнитивной и физической нагрузки. Поэтому IoBT должен помогать людям извлекать пользу из океана данных, принимая во внимание меняющиеся задачи миссии.

Естественно, противник не только будет физической угрозой для людей и IoBT, но и попытается проникнуть в саму сеть. Таким образом, сам IoBT станет «полем боя» с участием обороняющихся и атакующих. Как управлять рисками и снижать неопределенность в условиях враждебной среды?

Эти и другие вопросы обсуждались на совещании по стратегическому планированию, организованном Исследовательской лабораторией армии США в ноябре 2015 года с участием представителей научного сообщества, а также отраслевых и военных экспертов.

## УПРАВЛЕНИЕ И АДАПТАЦИЯ ИОВТ

Количество сетевых узлов IoBT для боевого отряда может быть на несколько порядков больше, чем когда-либо рассматривалось в рамках исследований. Особенно это проявится в ситуациях, когда участники боевых акций решат действовать сетевые устройства и каналы, им не принадлежащие. Например,



при ведении военной операции в мегаполисе можно воспользоваться доступными гражданскими устройствами Интернета вещей (рис. 2). В этом случае придется иметь дело с миллионом вещей на каждый квадратный километр.

С другой стороны, столь большой масштаб IoBT может быть полезным в теоретическом и практическом отношении. В частности, наличие огромного числа плотно размещенных датчиков позволяет решить проблему обеспечения постоянной доступности устройств, а чтобы это стало возможным, нужны теоретические исследования для выяснения степени детерминированности, доступной в рамках очень большого ансамбля вещей и данных.

IoBT будет также характеризоваться высокой гетерогенностью: локальные «сети вещей» состоят из множества коммерчески доступных устройств, а оборудование, которым люди будут пользоваться в боевых условиях, скорее всего, тоже будет основано на коммерческих разработках. Есть вероятность того, что и в будущем коммерческом Интернете вещей сохранится нынешний дефицит стандартов, в том числе из-за желания индивидуальных поставщиков контролировать рынок, поэтому военным нужна будет возможность быстрой адаптации и понадобятся соответствующие средства, чтобы пользоваться широким набором протоколов и коммуникационных технологий, поддерживаемых различными производителями.

В гетерогенной, высокодинамичной и труднопредсказуемой среде понадобятся новые способы быстрого обнаружения, выяснения характеристик доступных вещей и отслеживания их во времени и пространстве. В частности, у военных

при пользовании локальной сетью вещей населенного пункта не будет возможности делать достоверные предположения о поведении и характеристиках каких-либо частей IoBT — нужно, чтобы эти сведения собирались и обновлялись в ходе военной операции автоматически. Между тем сами люди тоже важные элементы IoBT, и, чтобы обеспечить их эффективную работу, нужно динамически распознавать, идентифицировать, характеризовать и предсказывать поведение солдат с обеих сторон и нейтральных гражданских лиц.

Масштаб, динамизм и высокий уровень сложности IoBT будут влиять на связь между вещами — для поиска каналов организации связи между огромным количеством разнородных, зачастую непредсказуемых вещей и управления этими каналами понадобятся совершенно новые подходы. Для непрерывного резервирования и перенастройки ресурсов сети связи потребуются высоконтеллектуальные средства автоматизации. Необходимо будет автоматически составлять и обновлять стратегии и правила обмена информацией, регламентирующие длительность и привилегии связи. Понадобятся высокомасштабируемые архитектуры и протоколы, а также надежные методы определения и подтверждения их свойств. В экстремальных ситуациях, когда в IoBT происходит катастрофический сбой, делающий его недоступным или ненадежным (например, в результате действий врача), автономные механизмы управления должны обеспечивать автоматическое восстановление, после которого можно продолжить работу, пусть и с деградацией функциональности.

Дополнительные сложности возникают в связи с ограничениями связи по времени. Какие-то коммуникации допустимо отложить на несколько часов, но для других типов связи (например, для передачи информации между датчиками и системами реагирования) нужна работа в режиме реального времени. К тому же доступность каналов связи будет сильно варьироваться. Прогнозируется, что через 30 лет в гражданском мире данные будут гарантированно проходить по беспроводной связи до надежных кабельных соединений дистанцию лишь в несколько метров, тогда как военным необходимы беспроводные каналы с охватом в десятки километров.

Картина общего состояния IoBT должна оперативно обновляться в автоматическом режиме, для чего понадобятся

новые методы извлечения необходимого объема сведений о сложных системах, основанные на регистрации относительно небольшого числа параметров.

Для эффективного управления IoBT нужно учитывать разнообразие его функций и применений. Некоторые из них ясны — например, военная логистика и распределенные вычисления. Другие будут порождены самим IoBT: его можно будет применять для нужд обнаружения, навигации, расчета времени, а также в качестве дополнения или замены GPS.

## ИЗВЛЕЧЕНИЕ ИНФОРМАЦИИ ИЗ ДАННЫХ IoT

Один из важнейших факторов, определяющих архитектуру IoBT, — ограниченная способность человека обрабатывать информацию [3]. Реагировать на все сведения, требующие внимания, в контексте IoBT невозможно. В сущности, один из ключевых рисков IoBT состоит в предоставлении информации, которая приведет к действиям с более негативными последствиями, чем если бы этой информации вообще не было.

Для предоставления полезных сведений, IoBT придется обрабатывать поистине беспрецедентные объемы сложных данных. При этом уровни абстракции, надежности и ценности такой информации — как генерируемой, так и потребляемой — сильно варьируются от устройства к устройству.

Вероятно, придется отойти от классической теории информации, в которой, в частности, рассматриваются ансамбли с равномерной плотностью вероятности, тогда как в IoBT будут протекать нелинейные и нестационарные динамические процессы, характеризующиеся неэргодической статистикой. Более того, при переносе информации между непохожими сетями могут возникать непредвиденные явления — например, изменение представления о ситуации вследствие обмена информацией между IoBT и социальными сетями, которыми пользуются участники боевых действий. Подобные процессы могут непредсказуемо влиять на способность людей контролировать IoBT и обмениваться с ним информацией.

Таким образом, необъятный массив данных IoBT нужно уменьшать до приемлемого уровня, выделяя действительно ценное содержание, готовое для передачи людям и «умным» вещам. По некоторым оценкам, объем информации придется уменьшить путем компрессии и консолидации в  $10^{15}$  раз. Один из путей реше-



Рис. 1. В рамках Интернета боевых вещей происходит обмен данными и взаимодействие разнообразных систем и устройств

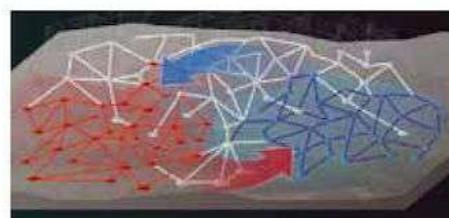


Рис. 2. Участники военных действий (красные и синие точки) осуществляют кибератаки — частично через гражданский Интернет вещей (серые узлы)

ния этой непростой задачи — дополнить IoBT многоуровневой иерархией информационных посредников, которые будут агрегировать, консолидировать, интерпретировать и пересыпать нужную информацию. Процесс консолидации нужно начинать на самом нижнем уровне — например, все вещи, генерирующие информацию, следует снабдить локальными средствами фильтрации, интерпретации и объединения данных. Такая система посредников может затруднить извлечение данных нижнего уровня, но, похоже, эту цену неизбежно придется заплатить, чтобы получать конструктивные сведения в приемлемом объеме.

Чтобы информационные посредники справлялись со своей задачей, они должны знать, какая именно информация полезна. Источником этих знаний могут быть процедуры планирования военной миссии и полевые учения, в рамках которых можно определить, какие именно сведения нужны людям и машинам. А для сохранения этих знаний нужен специальный язык выражения информационных потребностей для IoBT, доступный для машинной обработки, формальный, с широкой сферой применения и понятный военным. В ходе планирования и учений не вся нужная информация может быть получена — IoBT должен уметь самостоятельно выяснять, какие сведения

# Интернет вещей

необходимы для конкретной миссии и ее участников. Для этого потребуются подходы, основанные на машинном обучении и семантических знаниях.

Понадобятся также исполнимые модели IoT и окружающего его мира, которые обеспечат проверку, интерпретацию, консолидацию и оценку надежности информации. Помочь здесь может крупномасштабное моделирование. Сегодня исследования в области определения, автоматической генерации и динамического обновления таких крупномасштабных моделей находятся на самых ранних стадиях, а их конечным результатом должны стать эффективные решения для распределенного самомоделирования, самокалибровки и самопроверки IoT.

## ИСПОЛЬЗОВАНИЕ IoT ПРОТИВНИКОМ

Разумеется, противник будет осуществлять физические атаки против IoT, воздействовать на него направленной энергией, «глушить» каналы радиосвязи, разрушать волоконно-оптические кабели и уничтожать источники электроснабжения. Кроме того, он будет принимать меры, направленные на нарушение конфиденциальности, целостности и доступности информации IoT путем электронной прослушки и внедрения вредоносов. Последнее и, возможно, самое важное: противник будет проводить атаки на основе социальной инженерии. Люди — элементы IoT. Они уязвимы для обмана, в частности, для приемов, основанных на использовании личных и культурных пристрастий. Люди будут воздерживаться от использования IoT при подозрениях, что информация в нем ненадежна или что какие-то элементы сети контролируются противником. Похожие сомнения могут возникать и у систем искусственного интеллекта.

Один из основных приоритетов при этом — минимизация возможности получения врагом информации об IoT и людях, которые им пользуются. К IoT применимы многие меры, служащие для защиты традиционных сетей, но его исключительный масштаб, гетерогенность и плотность предоставляют дополнительные возможности защиты информации. Благодаря гигантскому количеству вещей в сети (особенно при использовании локальной гражданской сети вещей) есть возможность обеспечивать безопасность, просто отключая от IoT устройства, скомпрометированные врагом. Чтобы избежать прослушки

противником, можно, пользуясь изобилием «умных» вещей, внедрять в часть из них дезориентирующую информацию. Высокая плотность, сложность и разнообразие трафика сообщений затруднят противнику разведку иерархии управления и координации действий. Кроме того, при огромном количестве и плотности размещения вещей можно будет с меньшими затратами и более результативно ограничивать кибератаки врага путем создания крупных, внушающих доверие сетей-приманок, содержащих которые в обычном случае слишком накладно. Поддержка сети-ловушки в любом случае может обойтись дешевле, чем восполнение ущерба от разрушений, нанесенных врагом в результате успешной кибератаки.

Помимо попыток получить информацию, враг будет стремиться нарушить ее целостность путем модификации с помощью вредоносов, добавления собственных вещей к IoT, посредством порчи инфраструктуры и подстановки неверных сведений для регистрации датчиками. IoT должен бороться с этим с помощью механизмов распознавания аномалий, реагирующих на нестандартный трафик, необъяснимые изменения динамики или, наоборот, отсутствие ожидаемых событий. Обеспечить возможность распознавания аномалий помогут средства машинного обучения, способные справиться с гигантскими объемами данных, характерными для IoT. Непрерывный процесс обучения будет затратным с вычислительной точки зрения, и всегда есть опасность того, что противник сможет адаптироваться и развиваться быстрее, чем ваш самообучающийся механизм. Чтобы не позволить противнику вмешиваться в программную и аппаратную часть вещей IoT, нужны методы крупномасштабной «дактилоскопии» систем (например, на основе особенностей энергопотребления) и непрерывного общесетевого мониторинга соответствующих закономерностей. Понадобятся также средства непрерывного физического и информационного зондирования IoT, позволяющие распознать структуру сети и ее поведение, в том числе аномальное и подозрительное.

Процессы анализа штатного поведения систем и распознавания отклонений могут оказаться неэффективными в условиях дезориентирующих действий противника. Более того, обучение при этом может стать опасным оружием, направленным против того, кто его использует. IoT может запомнить какую-то законо-

мерность, подстроенную противником, после чего тот будет пытаться выполнять отвечающие ей действия, которые приведут к выгодному для него результату. Подобным образом, по сути, можно подделать любой индикатор нормы. И все же гигантские размеры и гетерогенность IoT способны помочь в борьбе с обмаными действиями с учетом того, что их масштаб, скорее всего, будет меньшим. В целом необходимо провести большой объем исследований в области методов противодействия дезориентации, ее выявления и защиты в сложных средах IoT. А поскольку сети IoT неизбежно будут соединяться с локальными гражданскими сетями вещей и, как следствие, с сетями противника, нужны будут инструменты, помогающие вести оборонительные операции в подобных «перепутанных» системах.

\*\*\*

Для Интернета боевых вещей могут потребоваться новые исследования в области теории игр при решении задач, для которых характерны огромное количество и разнообразие ходов и вариантов, практически неограниченные возможности зондирования, высокая сложность служебных функций и только частичная возможность наблюдения игровой доски. При теоретическом анализе обязательно нужно принимать во внимание вероятность обмана. В частности, требуются теоретические разработки, помогающие спрогнозировать уровень сложности дезориентирующих действий, при котором они могут достичь успеха. ■

## ЛИТЕРАТУРА

1. G.I. Seffers. Defense Department Awakens to Internet of Things. AFCEA.org, 1 Jan. 2015. URL: <http://www.afcea.org/content/?q=defense-department-awakens-internet-things> (дата обращения: 18.03.2017).
2. A. Kott, D.S. Alberts, C. Wang. Will Cybersecurity Dictate the Outcome of Future Wars? // IEEE Computer. — 2015. Vol. 48, № 12. — P. 98–101.
3. M. Kranz, P. Holleis, A. Schmidt. Embedded Interaction: Interacting with the Internet of Things // IEEE Internet Computing. — 2010. Vol. 14, № 2. — P. 46–53.

Александр Котт, Ананtram Свами, Брюс Вест ([alexander.kott1.civ](mailto:alexander.kott1.civ), [ananthram.swami.civ](mailto:ananthram.swami.civ), [bruce.j.west.civ@mail.mil](mailto:bruce.j.west.civ@mail.mil)) — научные сотрудники, Исследовательская лаборатория армии США.

# Уроки Мюнхена: миграция на свободное ПО



Стремительно растут темпы применения ПО с открытым кодом в различных областях, но в крупных организациях этот рост пока идет медленно. Не стал исключением и амбициозный проект LiMux, который завершился неудачей, — в начале 2017 года большинство членов городского совета Мюнхена проголосовало за установку на компьютерах городских служащих стандартной клиентской версии Windows.

*Ключевые слова: ПО с открытым кодом  
Keywords: LiMux, Open Source Software, Ubuntu*

Марио Силич, Андреа Бэк

**В** 1980-х годах, когда корпорация IBM практически монопольно господствовала в мире ИТ, многие руководители соответствующих отделов при выборе поставщика придерживались правила: «за покупку решений от IBM еще никого не увольняли». С тех пор положение дел изменилось. Во-первых, исчезла модель оплаты ПО как обычного товара [1], во-вторых, начало развиваться движение СПО (Open Source Software, OSS). Как следствие, появились новые экономические ориентиры и стратегические активы, которыми организации смогли пользоваться по своему выбору, — мир стал другим.

В наши дни происходит беспрецедентный рост применения СПО. Например, на SourceForge, популярной площадке хостинга открытого кода, насчитывается уже более 430 тыс. проектов, а доля Apache и nginx среди всех веб-серверов в мире составляет 54% [2]. Численность пользователей Android на различных устройствах приближается к миллиарду. Мир вступил в эпоху СПО — применение открытого кода для самых различных задач растет сегодня экспоненциальными темпами.

Если это так, почему то же самое не происходит в крупных корпорациях? Одно из возможных объяснений заключается в том, что, судя по статистике самой SourceForge, только 17% проектов миграции на СПО успешно развиваются, тогда как большинство бросается в самом

начале. И причина кроется в самих истоках модели СПО, основанной на принципах свободы выполнения, копирования, распространения, изучения, изменения и совершенствования ПО, а также выбора формы распространения версии. Однако эти принципы создают риски, которые при отсутствии грамотного управления чреваты для организаций дорогостоящими неудачами. Кроме того, на предприятиях могут оторваться СПО, не желая переходить на продукты с неопределенным будущим и размытым жизненным циклом.

Еще одна вероятная причина — опасения, связанные с использованием СПО. В 2012 году из-за вредоносного кода, внедренного в открытую систему веб-аналитики piwik, пострадало более 480 тыс. сайтов, а в 2014 году из-за бреши Heartbleed в криптографическом пакете OpenSSL уязвимыми оказались такие крупные сайты, как Google, Facebook и Yahoo. В результате во многих организациях по-прежнему не хотят внедрять СПО, поскольку ИТ-руководители ему не доверяют. По сути, зачастую дело не в реальном, а в воспринимаемом риске безопасности, что тем не менее влияет на принятие решения о внедрении.

На примере проекта внедрения Linux в администрации Мюнхена (точнее, операционной системы LiMux — дистрибутива, созданного на базе Ubuntu) рассмотрим принципы анализа связанных с СПО технологических рисков, а также разберем факторы, позволяющие снизить воспринимаемый уровень риска в процессе при-

ятия решений о подобных внедрениях. Анализ факторов риска (см. врезку) позволил составить список рекомендаций ИТ-директорам по эффективному освоению продуктов СПО.

## LINUX В МЮНХЕНЕ

Проект LiMux был начат администрацией Мюнхена в 2003 году после того, как городской совет решил отказаться от proprietарной системы и перейти на ПО с открытым кодом. Цель заключалась в том, чтобы заменить продукты Microsoft на свободные с открытым кодом. Толчком к этому послужило окончание поддержки Windows NT 4.0, которая использовалась в муниципалитете на более чем 14 тыс. настольных компьютеров. На всех также был установлен Microsoft Office 97 или Office 2000. Кроме того, применялось еще около 300 программных продуктов, включая браузеры, планировщики, клиенты для работы с мейнфреймами на операционной системе Siemens BS2000 и сетевые клиенты Novell. Наряду со всем этим в администрации применялось более 170 приложений, созданных специально для конкретных задач: макросы, формы и различные дополнения для приложений Microsoft Office. Во многих случаях эти дополнения были разработаны сторонними компаниями, связанными с администрацией Мюнхена контрактными обязательствами. Также использовался ряд серверных продуктов, в том числе СУБД Oracle для Unix и Adabas/Natural для BS2000, файловые серверы Novell Netware

Mario Silic, Andrea Back, Open Source Software Adoption Lessons from Linux in Munich. IT Pro, January/February 2017, IEEE Computer Society. All rights reserved.  
Reprinted with permission.

## Методология исследования

Летом 2014 года среди 115 ИТ-директоров было проведено анкетирование, в ходе которого участникам было предложено оценить по шкале от 0 до 10 значимость различных технологических рисков в процессе принятия решения о внедрении продукта с открытым кодом. Чтобы свести к минимуму сложности с ранжированием, участникам объяснили, что оценки имеют только относительное значение, то есть если поставить «7» фактору номер 1 и «3» фактору номер 2, это лишь означает, что первый важнее второго, а конкретные цифры роли не играют.

Затем из ряда источников была собрана информация о LiMux — проекте внедрения Linux в администрации Мюнхена. Сведения были получены из отраслевых СМИ посредством опроса участников и руководителей проекта LiMux, а также в социальных сетях.

и Sun PC-Netlink, сервер электронной почты Critical Path, календарные сервисы Oracle, факс-сервер Top Call и сервис каталогов X.500 Critical Path.

Руководство стояло перед выбором: либо перейти на новые версии продуктов Microsoft, либо полностью отказаться от proprietарных решений в пользу открытых. Выбор был сделан в пользу СПО, и через десять лет, в 2013 году, завершился проект по переводу почти 15 тыс. ПК на Linux<sup>1</sup>. Большая продолжительность аналогичных проектов может стать для многих препятствием, ведь принцип «время — деньги» зачастую становится определяющим в принятии ИТ-решений.

## ДЕСЯТИЛЕТНЯЯ ИСТОРИЯ

Ввиду высокой сложности имевшейся среди, процессов и требований, миграцию решено было проводить поэтапно. Главная задача LiMux состояла в обретении независимости от разработчиков и дистрибуторов коммерческого ПО, прежде всего от Microsoft. Другими словами, основной движущей силой инициативы были не деньги, а желание получить больше контроля над обновлениями при выходе очередных версий. Именно это предполагает модель СПО — свободу выбора. Со временем своего зарождения в 2001 году проект прошел несколько этапов:

- 2001—2003 — первоначальные планы;
- 2003 — официальный запуск;
- 2004 — официальный указ о начале миграции;
- 2005 — начало перевода первых ПК на LiMux;
- 2008 — немецкая служба технического контроля и надзора сертифицировала внедряемую систему как дружественную пользователю и соответствующую стандартам удобства;
- 2009 — пилотная фаза и начало отказа от Microsoft Office;

- 2012 — на открытое ПО переведено 12 тыс. ПК;
- 2013 — завершение проекта, на Ubuntu Linux и LibreOffice переведено 14,8 тыс. ПК.

Внедренное решение состояло из четырех основных компонентов: клиентской версии Linux с функциями автоматизированного развертывания и конфигурационного управления, офисного пакета, диспетчера шаблонов и форм WollMux, а также серверных компонентов.

На протяжении всей инициативы по внедрению проекта в организации шла нормальная деятельность, которая не прерывалась из-за процессов миграции.

## ТРУДНОСТИ И РИСКИ LIMUX

Первоначальные сложности отчасти были связаны с тем, что в 2002 году, когда был дан старт проекту, модель СПО еще не имела широкого признания в организациях и существенно отличалась от современного состояния. Если сегодня Android и Linux-серверы используются повсеместно, то 14 лет назад этого еще не было. Поэтому неудивительно, что в администрации Мюнхена после сравнительного анализа затрат на открытые и proprietарные решения не нашли финансовых оснований, оправдывающих переход на СПО. В итоге решение было принято по сугубо идеологическим соображениям — ради свободы выбора и избавления от привязки к поставщику.

## Юридические проблемы

В 2004 году возникла юридическая заминка, связанная с правовыми конфликтами из-за патентов, касающихся Linux и не только. Выяснилось, что к проекту могли иметь отношение более 50 спорных патентов, в том числе патент Amazon на покупку в один клик, а также относящиеся к XML, формату JPEG, файловой системе

CIFS и протоколу Server Message Block; большинство патентов принадлежали Microsoft. Миграцию приостановили, пока не был сделан вывод о том, что риск патентных конфликтов для проекта невысок.

## Трудности отхода от Office

В 22 департаментах муниципалитета использовалось более 21 тыс. шаблонов и порядка 900 макросов. На стадии оценки масштабов проекта выяснилось, что специализированные дополнения создавались большим числом разработчиков с использованием нескольких языков программирования. Таким образом, отход от Microsoft Office оказался одним из самых сложных этапов. В конечном счете в результате консолидации осталось только 12 тыс. шаблонов и 100 макросов, централизованно управляемых, контролируемых и документируемых.

## Сложности с обучением

Одна из трудностей была связана с тем, что новая система была крайне непривычной для пользователей: другой внешний вид, новые шаблоны, процессы и т. д. Изменить отношение пользователей к новшеству удалось путем убеждения ИТ-специалистов и сотрудников администрации в том, что в результате миграции их повседневная работа станет проще.

## Сложности с обеспечением устойчивого развития

Нужны были гарантии устойчивого развития, поскольку целью проекта была стандартизация ИТ-инфраструктуры, предусматривающая консолидацию всех существующих документов, процедур и процессов. Кроме того, необходимо было обеспечить самостоятельное управление версиями решений и гарантии прозрачности издержек.

## Риск отсутствия интероперабельности

Один из главных рисков был связан с интероперабельностью — в частности, с возможными проблемами при обмене документами и данными. Было принято решение об использовании в рамках LiMux стандарта Open Document Format для документов, редактируемых в офисных приложениях. А проблемы, возникавшие в процессе обмена данными, решались сообща с представителями той стороны, с которой происходила связь.

<sup>1</sup> Мюнхенские чиновники проголосовали за замену Linux на Windows, объясняя это, в частности, заботой об удобстве рядовых пользователей, привыкших к «стандартному» MS Office. Правда, в ходе обсуждений выяснилось, что проблемы, имеющиеся у ИТ-систем Мюнхена, искусственно раздувались, с тем чтобы склонить людей к отказу от LiMux. [www.computerworld.ru/articles/Myunhenskie-chinovniki-progolosovali-za-zamenu-Linux-na-Windows](http://www.computerworld.ru/articles/Myunhenskie-chinovniki-progolosovali-za-zamenu-Linux-na-Windows). — Прим. ред.

## Чрезмерная общая стоимость владения

Хотя целью внедрения LiMux не была экономия затрат, в ходе проекта выяснилось, что расходы на оплату услуг внешних консультантов оказались гораздо выше, чем ожидалось. Даже имел место спор по поводу общей стоимости владения проектом: официально было объявлено, что проект сэкономил городу десятки миллионов евро, тогда как Microsoft и HP опубликовали собственное исследование, которое показывало, что Мюнхен мог сэкономить 43,7 млн евро, если бы сохранил платформу Microsoft.

## Дефицит специалистов

Чтобы восполнить пробелы в компетенции собственного департамента ИТ, мюнхенской администрации пришлось нанимать специалистов по Linux и обращаться к услугам сторонних консультантов. Большинство собственных сотрудников муниципалитета специализировалось на продуктах Microsoft, в связи с чем пришлось делать значительные вложения в получение новых навыков в области СПО, чтобы обеспечить необходимый уровень поддержки пользователей. Понятно, что переход, скажем, с Windows NT на Windows 7 был бы менее затратен, чем с Windows NT на Linux.

## Сложности со стандартизацией

Проект LiMux в достаточной степени удалось сделать стандартом для всех департаментов, но на это пришлось потратить немало времени и сил. В частности, перед началом проекта в муниципалитете было выявлено 50 различных конфигураций Windows и пришлось обеспечить единство функциональности для всех соответствующих новых систем.

## ПЕРЕВОД ПОЛЬЗОВАТЕЛЕЙ НА СПО

Убедить служащих перейти на СПО было даже важнее, чем решить все технические проблемы. Есть два типа пользователей: одни отказываются работать с новым решением из-за мелочей, вплоть до непривычного цвета тех или иных элементов пользовательского интерфейса, а другие всегда готовы осваивать любую новую систему. В администрации большинство составляли пользователи первого типа.

Чтобы познакомить пользователей с дистрибутивом LiMux и основами работы в нем, проводились многочисленные собрания и презентации — проектная команда постаралась заранее в деталях

продемонстрировать служащим, что их ждет в дальнейшем. Некоторые сотрудники даже интересовались, смогут ли они пользоваться мышью, поскольку, как они слышали, в Linux есть только интерфейс командной строки. Эффективное просвещение пользователей по всем возникающим вопросам и глобальным задачам инициативы — обязательная часть подобных проектов, иначе освоение СПО неизбежно провалится.

## РЕКОМЕНДАЦИИ ПО ПРОЕКТАМ РАЗВЕРТЫВАНИЯ СПО

На какие факторы риска стоит обратить внимание?

### Поддержка — превыше всего

Поддержка на всех уровнях, возможно, самый важный фактор, влияющий на ход внедрения СПО. Без необходимой поддержки со стороны властей, высшего руководства, руководителей отделов, конечных пользователей, ИТ-департамента и т. д. внедрение СПО потерпит неудачу, как это произошло с проектом перехода на Linux в Вене. ИТ-директор должен позаботиться о получении полного одобрения от всех участников и заинтересованных лиц, иначе провал неизбежен.

### Тщательно оцените скрытые затраты

Если результативность проекта предполагается оценивать по окупаемости или совокупной стоимости владения, то будет велик риск провала. Нет полностью бесплатного ПО. Всегда нужно иметь в виду скрытые затраты, связанные с миграцией, — например, в связи с отсутствием технических знаний о внедряемых продуктах, из-за чего для сборки решения нередко приходится нанимать сторонних исполнителей. Но, несмотря на скрытые расходы, в долгосрочной перспективе ПО с открытым кодом будет малозатратным.

### Запаситесь временем

Сложные миграции действующих систем и платформ на СПО в большинстве случаев займут много времени. В рамках мюнхенского проекта большие сложности возникли, в частности, с переносом существующих макросов и шаблонов. Немало времени уйдет на анализ имеющихся систем и планирование миграции. При поэтапном подходе к проведению миграции, неизбежном в организациях со сложной структурой, обязательно надо мыслить долгосрочными категориями.

## Обеспечьте юридическую защиту и соблюдение нормативных требований

СПО — это «джунгли» из гигантского количества доступных продуктов, многие из которых не отвечают законодательным нормам и могут использовать патенты или фрагменты кода, не соответствующие открытой лицензии. ИТ-директор должен соблюдать предельную бдительность во всем, что касается юридических рисков и соблюдения нормативных требований, особенно если вспомнить многие недавние примеры, показавшие опасность использования продуктов с открытым кодом из-за нарушения законодательных норм. И здесь можно ориентироваться на подход, задействованный в рамках мюнхенского проекта, когда перед его началом было проведено обширное юридическое исследование.

### Удостоверьтесь в безопасности продукта

Любой программный продукт, входящий в экосистему организации, необходимо тщательно проверить, идет ли речь о небольшом одиночном приложении с открытым кодом или о сложном решении. Исходный код таких продуктов по определению доступен всем, в том числе злоумышленникам, которые могут его изменить, чтобы внедрить потенциально опасные программы. Проверка и контроль корректности продуктов СПО должны быть частью процесса внедрения.

\*\*\*

Стремительно растут темпы применения ПО с открытым кодом в различных областях, но в крупных организациях этот процесс пока идет медленно — руководителям, отвечающим за принятие решений, надо ориентироваться на приведенные рекомендации при оценке рисков, связанных с проектами СПО. ■

## ЛИТЕРАТУРА

1. G. Schryen. Is Open Source Security a Myth? // Comm. ACM. — 2011. Vol. 54, № 5. — P. 130–140.
2. August 2013 Web Server Survey. Netcraft, 2014. URL: <http://news.netcraft.com/archives/2013/08/09/august-2013-web-server-survey.html> (дата обращения: 01.03.2017).

*Марко Силич (mario.silic@unisg.ch) — научный сотрудник, Андреа Бэк (andrea.back@unisg.ch) — профессор, Институт управления информацией, Университет Санкт-Галлена (Швейцария).*

# Обучение программированию в эпоху технологических революций

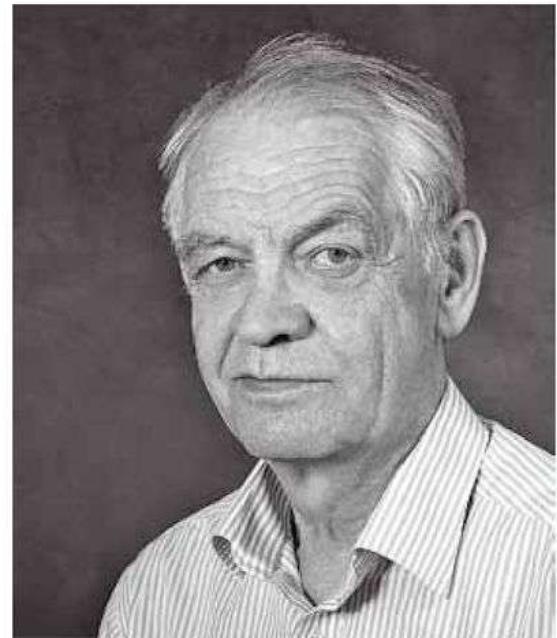
В ИТ происходит перманентная революция: изменяются технологии, инструменты, появляются принципиально новые решения. Как в этих условиях организовать процесс обучения, подготовить грамотных программистов, опирающихся на фундаментальные знания при выполнении прикладных разработок?

*Ключевые слова: свободное ПО, ИТ-образование и обучение  
Keywords: Open Source, IT education and training*

Виктор Иванников

С tremительное падение уровня вузовского образования в сфере ИТ, усугубляемое высокими темпами изменения технологий, резко обострило сегодня проблему организации учебного процесса подготовки высококвалифицированных программистов. Несмотря на то что российские программисты работают сегодня во всех ведущих компаниях мира, давно и с неизменным успехом применяющих разработанные у нас высокотехнологичные решения, над страной нависла угроза дефицита профессионалов в сфере ИТ. Где и как готовить программистов? Какое ПО использовать в учебном процессе? Свои соображения по этим вопросам высказал ведущий российский специалист в области системного программирования Виктор Петрович Иванников<sup>1</sup> в своем выступлении «Использование СПО в образовании» на Восьмой ежегодной конференции «Свободное программное обеспечение в высшей школе» (OSEDUCONF-2013), прошедшей в Переславле-Залесском 26–27 января 2013 года. Данная статья подготовлена редакцией с сохранением стилистики автора на основе этого выступления.

Мнение В.П. Иванникова, разработчика первой ОС для ЭВМ БЭСМ-6, предложившего средства организации параллельных процессов для отечественных многомашинных комплексов, одного из создателей отечественной научной шко-



лы системного программирования, по вопросам подготовки кадров сегодня важно как никогда. Именно В. П. Иванников предложил опробованную временем модель работы научных организаций — «образование через исследование», представляющую собой сплав образования, исследований и промышленных внедрений при тесном взаимодействии с ведущими учеными и исследовательскими лабораториями крупнейших игроков ИТ-индустрии. Ведущую роль в экосистеме из фундаментальных и прикладных исследований, индустриальных решений и системы подготовки кадров играет созданный им Институт системного программирования, получивший сегодня международное признание и авторитет у ведущих международных ИТ-компаний.

Статья сопровождается воспоминаниями соратников и учеников Виктора Петровича — представителей ИСП РАН и МГУ, принимающих сегодня непосредственное участие в подготовке программистов и реализации приведенных в статье идей. Надеюсь, это позволит получить более полное представление о многогранном характере В. П. Иванникова и задачах, которые ему приходилось решать.

— Дмитрий Волков,  
главный редактор, «Открытые системы.СУБД»

<sup>1</sup> Виктор Петрович Иванников (1940–2016) — выдающийся российский специалист в области вычислительной техники и системного программирования, создатель первой ОС для ЭВМ БЭСМ-6, руководитель проектов создания и внедрения систем автоматизации проектирования и программного обеспечения суперкомпьютеров. Основатель и первый директор Института системного программирования РАН с 1994 по 2014 год. В 1980 году удостоен звания лауреата Государственной премии СССР за разработку математического обеспечения системы АС-6. В 1984 году был избран членом-корреспондентом АН СССР, а в 2008 году — академиком РАН. Под непосредственным руководством В. П. Иванникова в ИСП РАН были, в частности, разработаны математические методы и алгоритмы анализа ПО, решающие задачи поиска уязвимостей и других дефектов программ на уровне лучших мировых систем; технологии анализа и извлечения семантики из больших массивов неструктурированных данных.

**К**ак хорошо было раньше преподавать: съездил Фибоначчи в Кордову или в Толедо, а там арабская десятичная позиционная система счисления, тогда как в Пизе была принята римская система — этот кошмар при выполнении арифметических действий. Вернувшись, он написал книгу о действиях арифметики, с задачами, которые и по сей день кочуют из учебника в учебник, — например, о популяции кроликов на необитаемом острове, откуда и пошла последовательность Фибоначчи. Книга на протяжении трех столетий служила в качестве удобного пособия в преподавании арифметики. Или более современное: Вейерштрасс упорядочил матанализ — ну, отличаются, конечно, современные курсы, но великий человек прекрасно все это сделал в первый раз... Сегодня же, особенно в ИТ, идет перманентная революция: технологии наращиваются, инструменты развиваются, появляются абсолютно новые вещи, и рассказывать студентам об этом становится все сложнее.

В конце 1960-х годов Никлаус Вирт проводил в Стенфорде курс по системному программированию, а в начале 70-х мне удалось прочитать ксерокопию его лекций. Это был сущий винегрет, чего там только не было: драйверы, семафоры Дейкстры, перевод из обычной инфиксной записи в польскую и т. п. Уже тогда остро всталась проблема упорядочения, причем студенты должны были освоить очень много инструментов, которые, как правило, относятся к открытому коду. Как их использовать? Причем не на уровне общих слов, а в конкретных учебных курсах для физтеха и ВМК: «Введение в алгоритмы», «Введение в архитектуру», курс по операционным системам для некоторых специальностей, в котором разбираются не только вопросы использования открытого кода, но и особенности работы с ним.

Мне много приходилось читать статей — к сожалению, не отечественных, а американских — о проблеме первого языка, на котором человек может написать программу. Я давний сторонник Паскаля, потому что, несмотря на несуразность языка, все-таки Вирт, а он умница, создавал свой язык совсем не для разработки программ, а для того, чтобы на нем учились программировать. Там строгая типизация, а не та вольница, с которой привык работать профессиональный программист. Но потом я столкнулся с такой вещью: и на ВМК, и на физтехе у ребят буквально создается путаница в головах. В первом семестре им дают фундаментальные алгоритмы, почти все можно было писать на Паскале. Мне

хотелось, конечно, одну лекцию посвятить алгоритмам запросов к памяти, а в Паскале, ввиду строгой типизации, все можно написать, но коряво же будет выглядеть — нужно понятие адреса и пр. В результате студенты не знают, как происходит динамическое распределение памяти, а в следующем семестре они изучают ассемблер. А на физтехе еще и макроассемблер! Ну зачем еще макро?! И таким образом совсем сбивают студента. А потом начинается следующий семестр, рассказывают про Unix, а Unix и Си — это неразрывные вещи, и ребята уже полностью дезориентированы. И вот возникла идея сквозного прохода по курсам — использовать язык Си для разных вещей.

Итак, курс «Введение в алгоритмы»: изучение языка Си; вводный курс в алгоритмы; структуры и типы данных; таблицы; хеширование; деревья; очереди; стек; машинная арифметика, не имеющая отношения к математике, — не выполняются законы коммутативности и ассоциативности, но студент должен все это осознать. И конечно, нужны какие-то понятия об алгорит-

мической сложности — мы живем в мире эквивалентных алгоритмов (а их синонимы — «программы»), и нужно уметь их взаимно оценивать. Здесь же рассказывается о машине Тьюринга, что такое неразрешимость и прочие обычные понятия, по которым имеется много учебников.

Принципиальный вопрос курса «Введение в архитектуру» — как рассказывать, собственно, про архитектуру? Наверное, близко к языку ассемблера, однако стили программирования на языке высокого уровня и на ассемблере разные, другой менталитет. Понятно, что некоторым придется реально писать на ассемблере, но с инструкциями, тем не менее, уметь работать нужно. Идея заключается в том, что нужно сконцентрироваться на мэппинге — студент должен видеть две программы: на Си и эквивалентную на ассемблере. Должен понимать обе и уметь вручную восстанавливать конструкции языка Си по ассемблерному коду. Студенту нужно именно понимать, а не научиться в совершенстве писать ассемблерные программы, не глядя в руководство. Он должен ори-

## Дело прежде всего

Под руководством Виктора Петровича я начал работать в 1996 году, когда поступил в аспирантуру ИСП РАН. Теперь кажется, что он был рядом всю мою сознательную жизнь, учителем, наставником, научным руководителем, а с какого-то времени и другом.

Для меня, как и для многих, он был живой легендой, однако работать с ним было непросто. Виктор Петрович всегда предъявлял очень высокие требования, ставя дело на первое место, а разговоры с ним на любые темы: о Вселенной, о Древнем мире, о книгах — часто скатывались к обсуждению проблем института. После бурных дискуссий на семинарах он мог позвонить в любое время, и мы продолжали беседу.

Каждая из созданных Виктором Петровичем операционных систем разрабатывалась на основе самых передовых научных достижений своего времени, а главное — все шесть ОС были запущены в промышленную эксплуатацию. Научная школа Виктора Петровича показала свою жизнеспособность и эффективность: в институте под его прямым руководством были разработаны технологии мирового уровня, успешно конкурирующие с лучшими мировыми аналогами. Например, статический анализатор кода Svac, развиваемый с 2000-х годов, сегодня взят на вооружение в компании Samsung. В последнее время Виктор Петрович руководил разработкой отечественной ОС для авионики. Ему было 76 лет, но он в полную силу участвовал во всех разработках, показывая нам пример.

В любом обществе есть люди, на которых равняются, их уважают за ответственность, нацеленность на результат, за волевые, моральные качества — именно таким был Виктор Петрович, всегда все делавший по совести и ставший для нас не только авторитетом в научном плане, но и мерилом порядочности. Виктор Петрович говорил: «Хороший человек — не профессия. Но если есть хороший человек и он хочет стать профессионалом, из него всегда можно сделать профессионала».

Благодаря созданной атмосфере высокого профессионализма, абсолютной честности и уважения друг к другу, ему удавалось привлекать в институт пассионарную молодежь, которая сейчас составляет у нас 80% сотрудников. Это позволило решить самую трудную задачу — преемственность поколений.

Трепетное отношение к своим ученикам я почувствовал и на себе — при таком отношении никакой уровень ответственности не казался завышенным. Виктор Петрович часто создавал ситуации, где нам казалось, что мы все придумали сами, чтобы могли гордиться своей работой и хотели двигаться дальше. Мы доверяли ему безусловно, хотя сам он не любил говорить о себе и был очень скромным.

— Арутюн Аветисян,  
член-корреспондент РАН, директор ИСП РАН

## Цели и средства

Под руководством Виктора Петровича я работал с начала 1984 года. В то время мы разрабатывали системы программирования и операционную систему для нового советского суперкомпьютера, уже через три года создав два ассемблера, ОС и пять компиляторов. В процессе этой работы сформировался коллектив, на базе которого в 1994 году и был организован ИСП, однако ему сразу пришлось столкнуться с испытаниями: финансирование иссякло, ведущие программисты уехали за рубеж. Тем не менее Виктор Петрович сумел заключить контракты на разработку ПО с крупными компаниями, например с Hewlett Packard. И с конца 90-х годов институт уже работал в нормальном режиме: появились деньги, большая часть которых шла на увеличение зарплаты сотрудников, и у них пропал стимул уезжать, текучка кадров прекратилась, студенты и аспиранты стали оставаться в ИСП после учебы, а институт получил признание как один из лучших институтов академии наук. Со многими компаниями, с которыми начали сотрудничать в 90-е, мы до сих пор продолжаем работать.

Виктор Петрович был не только выдающимся ученым, но, самое главное, честным и порядочным человеком, одним из немногих, кто понимал, что деньги — не сама цель, а лишь одно из средств достижения целей. А цели у него всегда были высокие и благородные. Пока я жив, буду постоянно вспоминать Виктора Петровича как лучшего из людей, с которыми мне посчастливилось дружить и работать.

— Сергей Гайсарян,  
заведующий отделом компиляторных технологий ИСП РАН

ентированность в семантике программы на ассемблере и уметь переписать ее на Си. Хотя отчасти можно это сделать автоматически, но в этом случае код на Си будет очень напоминать ассемблер.

Однако здесь есть тонкости: если во времена моей молодости архитектура компьютеров была достаточно простая, то сейчас, используя различные подходы работы с блоками памяти, разные шаги обработки в цепи массивов, разную иерархию кэшей и т. д., можно достигать производительности, отличающейся на порядки. К сожалению, нынешние студенты, да и аспиранты, этого не понимают, но именно это хотелось бы заложить как можно раньше, чтобы студенты осознали, что такое компьютер. Важно, чтобы студент не просто услышал различные понятия, связанные с термином «кэш», а понял, как он работает, что такое промах кэша. Цель курса как раз в этом. Конечно, есть еще многоядерность, когерентность кэшей и др., что можно оставить на аспирантуру, помня, что на этом-то и строится все распараллеливание.

Важно отметить, что даже лекционные курсы должны читать молодые специалисты — разница в возрасте не должна быть слишком большой, а когда курсы по ИТ читают восьмидесятилетние, это ужасно. Лекции — это верхушка айсберга курса, а основная работа идет на семинарах и в компьютерном классе. На ВМК — это шесть групп, и чтобы процесс был эффективным, нужно по два преподавателя на группу. Конечно, надо предусмотреть много самостоятельной работы, но, помимо прочего, есть индивидуальная работа — это не только решение задачи с автоматическим тестированием результатов, но и обучение под контролем преподавателя.

В третьем семестре на ВМК читается курс по операционным системам, однако

## Кадры для науки

Виктор Петрович пришел к нам в МГУ в 1993 году на кафедру, которую до него возглавлял патриарх отечественного программирования М. Р. Шура-Бура. За почти полувековой период существования факультета ВМИК это единственный пока случай, когда новый молодой руководитель и прежний долгие годы успешно и плодотворно работали вместе. Профессура кафедры, знающая Виктора Петровича как высококлассного специалиста иченого, безусловно, одобрила его приход. Работа со студентами, лекции, экзамены — это для Виктора Петровича было «святое», и при всей своей занятости он никогда не отменял лекции. В непростые 90-е годы Виктор Петрович, используя свои связи с зарубежными учеными, обеспечил безвозмездную поставку 100 персональных компьютеров из США, что позволило факультету поднять образовательный процесс на качественно новый уровень. За время работы на кафедре Виктор Петрович вырастил несколько поколений программистов, которыми по праву может гордиться не только Россия.

Став завкафедрой МГУ, Виктор Петрович одновременно занимался созданием Института системного программирования и очень гордился тем, что именно в Татьянин день был подписан приказ о создании ИСП РАН. Виктор Петрович очень много внимания уделял тому, чтобы молодые специалисты плодотворно работали, и многие талантливые ребята остались в науке исключительно благодаря его усилиям. К нему можно было обратиться с любой просьбой и вопросом, несмотря на его строгость и требовательность, — вершины в науке, которых он достиг, не заслоняли для Виктора Петровича проблем любого коллеги.

— Людмила Корухова,  
заслуженный преподаватель МГУ им. М. В. Ломоносова

## Преподавать должны профессионалы

Мне посчастливилось вместе с Виктором Петровичем работать над подготовкой нового учебного курса, тема которого ему была близка, как никакая другая. Началась эта история довольно неожиданно — с участия в приеме вступительных экзаменов в аспирантуру. Ответы выпускников университета на вопросы о работе операционных систем и обработке прерываний, скажем мягко, нельзя было назвать уверенными, что не оставило равнодушным Виктора Петровича. Почему талантливые ребята выходят из университета с такими пробелами в понимании базовых областей системного программирования? Только прочувствовав проблемы, можно полноценно освоить материал, поэтому решение созрело быстро: необходим курс по ядру операционных систем с акцентом на практической работе.

Я в тот момент уже около 10 лет участвовал в проведении курса «Методы формальной спецификации и верификации программ». Вызвав меня к себе в кабинет, Виктор Петрович был

твер: «Преподавать предмет должны те, кто профессионально работает в этой области, а у нас с операционными системами больше всего работаешь ты!» Действительно, в наших проектах по верификации модулей ядра ОС Linux и тестированию ОС РВ мне приходилось разбираться во многих деталях, готовить исправления выявленных ошибок в ядре Linux, поэтому взорвать было нечего. Так стартовала работа по подготовке нового курса. Четыре месяца регулярных обсуждений, изучения курсов других университетов, учебных операционных систем, собственные истории из практики Виктора Петровича и пр. Но, наверное, самое ценное, что я вынес из общения с Виктором Петровичем, — это его очень теплое, но требовательное отношение ко всем ребятам, что в конечном счете способствовало более полному раскрытию талантов молодых специалистов.

— Алексей Хорошилов,  
директор Центра верификации ОС Linux ИСП РАН

на госэкзаменах ребята не понимают, что такое прерывания, и абсолютно не чувствуют синхронизацию. Если бы студенты за время обучения написали хотя бы один драйвер, где поработали бы с привилегированными командами, то сразу почувствовали бы, что такое запрещение и разрешение прерываний, какие там могут быть эффекты. В этом же курсе обязательно должно быть предусмотрено написание простейших драйверов, в том числе драйвер дисковой и виртуальной памяти — здесь есть тонкие моменты в работе с таблицами и переключателями. Здесь же потребуется написать планировщик, причем надо, чтобы была работа с запрещенными командами, с прерываниями, это управление виртуальной памятью, управление драйверами, переключение между задачами. Цель — научить студента все это делать очень аккуратно.

Главная задача — чтобы студент сам что-то сделал в операционной системе. В идеале — если бы в течение семестра коллективно удалось создать некоторую операционную систему: тривиальное управление внешней памятью, управление задачами, переключение процессора, может быть, разделение времени. В западных университетах это практикуется.

Лично я написал шесть операционных систем и представляю, как просто и быстро написать ОС, но для учебных целей она должна быть компактная. Можно взять за основу систему JOS из МТИ, тем более что лицензия относится к классу BSD. Операционная система JOS — это около 8 тыс. строк на Си и 400 строк на ассемблере. Эмулятор QEMU, хотя есть и другие, например Valgrind, но QEMU хорош тем, что исполняет привилегированные команды и можно не прикрываться библиотечными вызовами. Отладчик gdb, компилятор gcc.

Данный курс нужно уложить в один семестр, и здесь кроется другая серьезная проблема — дефицит специалистов в сфере технологий компиляции, знающих предмет не на уровне учебников, а на уровне практиков, разбирающихся в коде. Люди, которые преподают курс, сильно загружены, обычно они еще работают над своими кандидатскими диссертациями, и если растянуть этот курс на год, то им просто придется переквалифицироваться в преподаватели. Смысль заключается в том, чтобы студентов обучали профессионалы в своей области, а это предполагает обязательное участие и студентов, и преподавателей в реальных проектах.

Отдельного разговора заслуживают компиляторы, причем обсуждать в курсе



## Создатель научной школы системного программирования

С Виктором Петровичем нас связывают долгая совместная работа и дружба. Мы познакомились в конце 1968 года в ИТМиВТ. Когда был создан Институт системного программирования, Виктор Петрович пригласил меня, и 18 лет я был заместителем директора института по науке, приходилось решать и разные административно-хозяйственные вопросы.

С первых дней работы нового института Виктор Петрович столкнулся с массой проблем, начиная с необходимости восстановления полуразрушенного здания, доставшегося в наследство от Института проблем кибернетики, для создания нормальных условий труда сотрудников, поиска средств финансирования исследовательских подразделений и создания материально-технической базы, и заканчивая вопросами воспроизводства высококвалифицированных кадров. Оглядываясь назад, можно констатировать, что лишь несгибаемая воля и неукротимая энергия Виктора Петровича позволили превратить институт в ведущее в нашей стране научное учреждение в области академических исследований и технологических разработок в системном программировании.

С ним было довольно легко работать, хотя руководителем он был достаточно жестким. Сам он был очень дисциплинированным человеком и требовал этого от своих подчиненных. Он всегда был готов оказывать необходимую помощь и поддержку при решении возникающих проблем.

Виктор Петрович был всесторонне эрудированным человеком. Любил литературу, живопись. Собрал довольно большую коллекцию редких минералов. В быту он был очень скромен. Уделял внимание спорту и был отчаянным грибником.

Он постоянно учился. Читал и просматривал практически все приходящие в институт периодические издания, прекрасно разбирался во всех тонкостях современных информационных технологий.

Основная заслуга Виктора Петровича, с моей точки зрения, — создание научной школы системного программирования, включающей десятки докторов и кандидатов наук, и коллектива единомышленников, способного решать самые актуальные задачи современного системного программирования и продолжать начатое им дело.

— Виктор Шнитман,  
заведующий отделом ИСП РАН

надо их внутреннее устройство и архитектуру, а не внешний интерфейс. Для целей обучения нужен достаточно компактный открытый код, чтобы он позволял вставлять различные плагины, и тут есть серьезная проблема с выбором. Например,

gcc — это сейчас несколько компиляторов: фронтенды для каждого из входных языков программирования и два внутренних представления (только планировщиков кода там четыре штуки). Один из планировщиков мы по контракту с HP делали

для микропроцессора Itanium, в котором много регистров, поэтому нельзя проводить их распределение только на базовом блоке, на линейном участке, а необходимо захватывать переходы, что означает спекулятивное выполнение и вероятностную оценку пути. Для того чтобы нашей команде углубиться в среду gcc, в свое время потребовался год. Ясно, что, если бы студенты были знакомы с этим компилятором еще в институте, такие работы выполнялись бы быстрее.

Другая вещь — LLVM (Low Level Virtual Machine), инфраструктура генерации оптимизированного машинного кода для различных платформ, используемая, в частности, в компаниях Adobe, Google и Apple, последняя из которых финансирует сообщество развития этой системы. Здесь речь идет о другом представлении, отличном от стековой машины, с более современной архитектурой, допускающей простую интеграцию, которой в gcc, ввиду древности этого компилятора, нет. Поэтому для целей обучения LLVM предпочтительнее: на этой архитектуре можно делать, например, компиляцию just-in-time, что иногда полезно для получения оптимального кода, когда в статике заранее неясно, как оптимизировать доступ к памяти и размещать регистры. Если заранее не знать размеры памяти, то будут сплошные промахи в кэше. Для LLVM имеются всевозможные API, кодогенерация для GPU разных производителей, и даже есть бинарная компиляция из кода одного процессора в код другого. Сама идея бинарной компиляции возникла в начале 90-х и впервые была реализована для знаменитого процессора DEC Alpha при решении проблемы переноса унаследованных приложений с VAX VMS на платформу Alpha, причем без деградации производительности. Аналогичная проблема сегодня имеется и для «Эльбруса»: архитектура и кристалл есть, но кто будет писать приложения для этой платформы, учитывая, кроме того, что большинство имеющихся приложений коммерческие? Выход только один — трансляция бинарного кода.

Курс по компиляторам, опять же семестровый, требуется разрабатывать самостоятельно — я не вижу явных примеров зарубежных университетов, курсы которых можно было бы взять за образец.

Отдельно надо сказать про прикладные пакеты и среды типа OpenFOAM. Сегодня в стране сложилась катастрофическая ситуация. Все отечественные производители: «КамАЗ», «Рыбинские моторы», «Сухой», «Микоян» и другие, даже нефтянка — «сидят» на иностранных расчетных пакетах.



## Ошибка начинающего дизайнера

Мы начали тесно общаться с Виктором Петровичем, когда я поступил в аспирантуру факультета ВМК и стал участвовать в проектах с компанией Nortel Networks. Особенно запомнились семинары в кабинете директора, на которых Виктор Петрович старался вникнуть во все детали, даже если не был непосредственным руководителем проекта. Когда начиналась работа по реализации нового планировщика для компилятора gcc, мне нужно было сделать для него доклад, в ходе подготовки к которому стали понятны многие нюансы, которые необходимо было учесть. Запомнился первый разнос, который я получил от Виктора Петровича за ошибку проектирования — результаты работы первой версии компилятора варьировались от запуска к запуску. Причина была в неправильно выбранных структурах данных, а в ИСП родилось популярное теперь выражение «ошибка начинающего дизайнера». Много позже, когда я сам стал вести проекты, я понял, что тогда директор внушение сделал исполнителям весьма мягко и корректно, а руководители, напротив, получили «по полной». Молодых сотрудников Виктор Петрович очень берег.

Виктор Петрович был очень живым и увлекающимся человеком, интересующимся не только программированием, — для него было естественно сослаться в лекции на Гнедича или «Илиаду», поинтересоваться у молодого сотрудника, что тому в школе ставили за сочинение. А прочитав черновик служебной записи в Президиум РАН, он мог посоветовать сотруднику брать пример с Флобера, дважды не повторявшего на странице одно и то же слово. Мы и сегодня часто обсуждаем важные проблемы в кабинете Виктора Петровича, и сложные решения принимаются как-то легче и спокойнее.

— Андрей Белеванцев,  
руководитель направления анализа и оптимизации программ ИСП РАН

Однако каждая ведущая западная компания (Boeing, Airbus, Intel, Schlumberger) имеет у себя на вооружении собственные системы расчета. Сегодня качество проектирования определяется качеством пакетов CAD/CAM/CAE, а то, что предлагается на открытом рынке, — это ширпотреб. Не имея доступа к таким системам и при отсутствии разработки собственных, наша экономика в результате автоматически отстает лет на десять, несмотря на то что средства на приобретение таких западных пакетов тратятся огромные. Есть, конечно, отечественные программы, например FlowVision, но смешно их сравнивать с западными, в развитие которых вложены миллиарды. Выход — открытый код, что уже многие осознали. Например, 80% всех вычислительных экспериментов в Audi выполняются на OpenFOAM.

В OpenFOAM для разных задач имеется больше сотни специализированных программ — решателей, которые можно модифицировать под конкретную область или же создать свои с нуля. Конечно, можно заказать решатель у той же ANSYS, но хватит ли у вас средств на его оплату, при том что специалисты у нас умные, сами способны организовать вычислительное моделирование. На «КамАЗе» в свое время использовали продукты ANSYS, но там не оказалось нужного решателя для моделирования подрыва заряда тротила под дном автомобиля. Тогда все сделали сами, написали программу, провели вычислительный эксперимент, результаты которого совпали с данными натурных испытаний. Может быть, эту программу следовало использовать шире, например, автобусы так проверять, но программа, по сути, так и осталась одноразовой, а нужно было, чтобы она была интегрирована в среду типа OpenFOAM и была доступна.

Какой бы пакет вы ни взяли, он все равно не обеспечит полноты — так или иначе будут возникать модельные задачи. Можно делать свой решатель и сохранять его в общей копилке, доступной всем желающим, а можно, решив задачу и получив результат, отложить в сторону в надежде на то, что когда-нибудь какой-нибудь студент это откуда-то возьмет. Но тут есть одна проблема. Мы провели несколько конференций и тренингов, в том числе совместно с немцами, которые активно используют OpenFOAM. И сделали такой вывод: человек должен работать внутри этой системы, а для этого он должен прослушать курсы по урматам, по вычислительным методам, должен иметь какое-то представление о параллельном программировании, уметь

## Романтик и реалист

Виктор Петрович уважал и высоко ценил талантливых и порядочных людей. Будучи сам таким же, он не раз говорил и отмечал в своих эссе, в частности о работе в ИТМ и ВТ и о своем учителе Льве Николаевиче Королеве, что счастлив жить именно среди таких людей. В его характере успешно сочетались качества романтика и реалиста в науке.

Я работал рядом с Виктором Петровичем со дня его прихода в ИТМ и ВТ — могу свидетельствовать о его постоянно сердечном отношении к делу и товарищам. Даже в случае неудачно складывающейся ситуации в работе, он, всегда прекрасно понимая суть дела, никогда не допускал несправедливых упреков.

Мыслил Виктор Петрович быстро, с ходу «определял» ситуацию. Приходилось, например, наблюдать, как, будучи оппонентом по докторской, он почти мгновенно готовил содержательный отзыв, что, кстати, отмечали и студенты МФТИ: «Лекции Иванникова слушать непросто, но шарит он немерено...»

О студентах Виктор Петрович заботился постоянно: по окончании учебы решительно говорил им, что надо идти работать «туда, где есть чему учиться и есть кому учить». Именно Виктору Петровичу принадлежит идея организации послевузовского обучения специалистов экстра-класса в процессе выполнения ими концептуально значимых проектов по созданию новых технологий разработки программного обеспечения вычислительных систем.

— Александр Томилин,  
главный научный сотрудник ИСП РАН

## Идеи, меняющие представление о проблеме

Виктор Петрович любил работать со студентами. Когда я еще учился на первом курсе ВМК МГУ, он читал курс по структурам данных и алгоритмам. Первая пара в субботу собирала весь поток, и после лекции всегда было сложно пробиться к нему через толпу студентов с вопросами. Операционные системы, которые разрабатывал Виктор Петрович, эксплуатировались на лучших для своего времени компьютерах и летали в космос — этот опыт очень чувствовался на его лекциях. Хотя сам он считал, что преподавать программирование должны люди, разрабатывающие промышленное программное обеспечение, — именно так, из первых рук студенты могут постичь мастерство разработки современного ПО. Эта идея воплотилась в модели работы ИСП РАН — сейчас большинство сотрудников руководят научной работой студентов, многие читают лекции и ведут семинары в МФТИ, МГУ и ВШЭ. Кстати, именно благодаря Виктору Петровичу при преподавании алгоритмов на ВМК МГУ стали использовать промышленный язык Си вместо академичного Паскаля.

Как человек и руководитель он был очень разносторонним: свободно говорил по-английски, у него на столе можно было увидеть разные книги от классической литературы до истории Кореи. В его кабинете можно было встретить не только сотрудников, но и студентов младших курсов базовых кафедр, которым требовался его совет. Казалось, что на любой вопрос он может либо ответить сам, либо дать прочитать книгу. Всегда можно было прийти к нему и обсудить сложную проблему — как научную, так и управленческую. Часто он сразу выдавал неожиданное решение. Иногда брал паузу, а потом мог позвонить поздно вечером или рано утром, чтобы обсудить идею, которая пришла ему в голову. Зачастую эта идея меняла все представление о проблеме и в дальнейшем приводила к успешному решению.

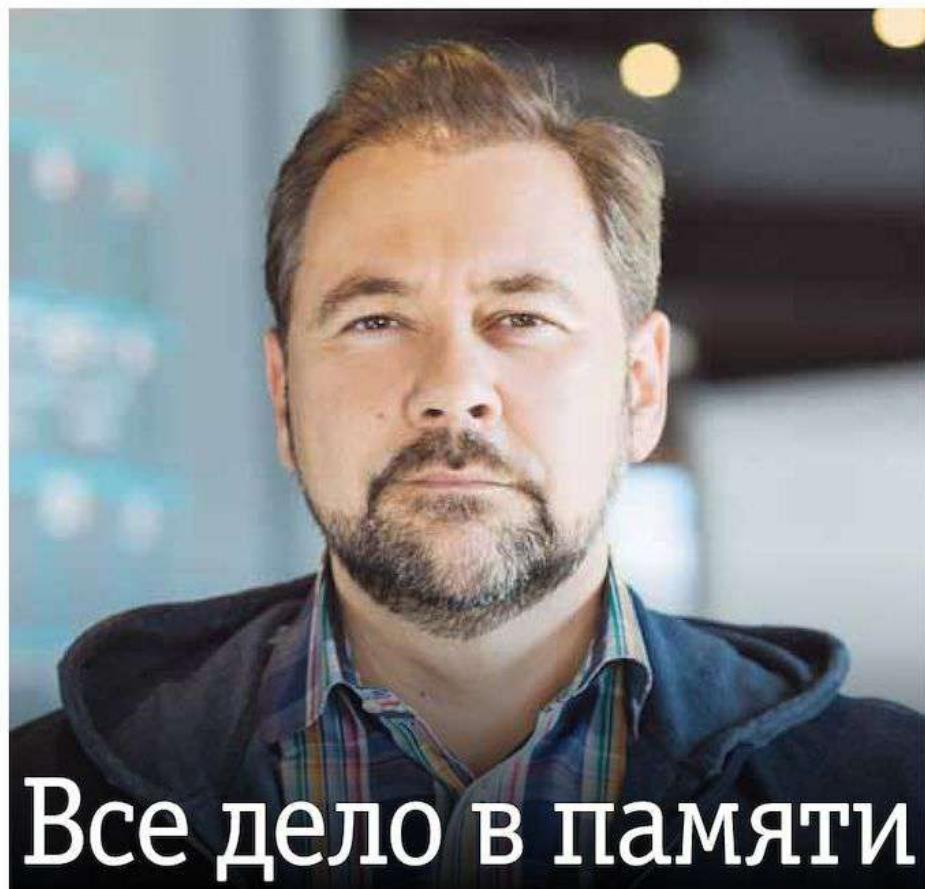
— Денис Турдаков,  
заведующий отделом информационных систем ИСП РАН

писать параллельные программы, наверное, он должен знать механику — он же что-то описывает, какие-то физические модели.

Все эти инструменты достаточно сложны даже в смысле API, поэтому, конечно, нужно пройти серьезный тренинг по пакетам. Я однажды был на научно-техническом совете «Рыбинских моторов», и там выступал в том числе главный технолог или главный инженер. Они используют ANSYS, и, конечно, у них идет производственный процесс. И когда я сказал про OpenFOAM, он говорит: «Вы поймите такую вещь: у нас

же процесс, мы попробовали OpenFOAM, но не пошло». Потому что OpenFOAM — это не только пользовательский интерфейс, нужно влезть вовнутрь, чтобы получить другое качество! При этом автоматизация пакетов и пользовательские интерфейсы — это целая отрасль, которой успешно занимаются отдельные компании, например в Германии и в рамках Open Source.

Все это относится и к открытому коду в целом: используют широко, но очень редко, когда кто-то «влезает внутрь», к сожалению. ■



## Все дело в памяти

Технологии, обеспечивающие высокую скорость обработки и масштабируемость хранения данных в памяти, не так просты, как кажется, хотя бы по причине разнообразия задач и данных.

*Ключевые слова: гибридные технологии, направления развития СУБД, память*  
*Keywords: database development future, GridGain, hybrid technology, memory*

Наталья Дубова

О компании GridGain Systems заговорили в 2015 году, после того как стало известно, что Сбербанк планирует масштабное внедрение технологий обработки в памяти GridGain In-Memory Data Fabric и становится инвестором компании. Никита Иванов, основатель и технический директор GridGain Systems, родился и вырос в Ленинграде, окончил Балтийский государственный технический университет «Военмех», а с середины 90-х годов живет и работает в США. У него имеется более чем двадцатилетний опыт разработки программного обеспечения для высокопроизводительных систем и платформ промежуточного слоя. Он один из первопроходцев применения Java для разработки серверного промежуточного

ПО. В 2005 году Иванов вместе с коллегами начал работу над распределенной технологией обработки данных в памяти, которая и положила начало продукту GridGain In-Memory Data Fabric. В 2015-м по результатам тендера решение GridGain было выбрано в качестве инфраструктурной платформы для проектов Сбербанка, и с прошлого года идет внедрение системы в производственную среду этой крупнейшей в мире финансовой организации.

В интервью журналу «Открытые системы» Никита Иванов рассказал об истории и перспективах своей компании и о том, почему технологиям in-memory альтернативы нет.

**Как была создана компания GridGain?**  
 Как и многие проекты в области инфраструктурного ПО, проект, который лег

в основу технологий GridGain, начинался в рамках сообщества Open Source — сегодня это проект Apache Ignite, развивающийся с середины 2000-х. Компания GridGain Systems была создана в 2010 году вокруг этого проекта, когда были привлечены первые инвестиции.

Идея вычислений в памяти (in-memory computing) не нова — она появилась еще в 1960-е годы, с началом применения первых внешних накопителей на лентах. Доступ к ним был на несколько порядков медленнее, чем доступ к данным в памяти, поэтому сразу возник вопрос: почему бы не держать часто используемые данные в памяти, там же, где выполняются программы? С тех пор практически все системы по хранению и обработке данных имели возможность кэширования. В первых реализациях in-memory computing было совсем мало памяти, она была безумно дорогой, но с годами ситуация изменилась: падение цен на память, широкое распространение 64-битных процессоров и всплеск объемов данных — все это определило возможность современных реализаций in-memory computing. Наличие 64-битного процессора позволяет адресовать доступ к множеству данных на одном узле, а если соединить эти узлы в кластер, то появляется возможность работать с очень большими объемами данных. И самое главное, подоспели гражданские приложения, которые нуждаются в таком подходе к обработке данных, — в течение 20 лет это были в основном специализированные задачи военного назначения.

Сегодня in-memory computing становится для большинства современных приложений фактическим стандартом. И наш проект — один из тех, что возникли на последней волне развития технологии обработки в памяти.

Интересно также посмотреть на эту технологию с точки зрения эволюции систем хранения данных. Развитие шло от лент к жестким дискам, затем к флеш-памяти и сейчас — к хранению в памяти. К 2018 году прогнозируют активное развитие энергонезависимой памяти (non-volatile memory, NVM) и фактически хранение данных в памяти — это последний рубеж в эволюции хранения для компьютеров архитектуры фон Неймана. До тех пор пока мы кардинально не изменим эту архитектуру (например, перейдем к световым или биокомпьютерам), существенных прорывов в подходах к хранению уже не произойдет. Именно поэтому большинство аналитиков считают, что in-memory computing — это не просто очередная технология хране-

ния, такая как флеш, а основной способ для работы с данными на долгое время.

### **В обозримом будущем альтернатива этому подходу не появится?**

Да, для многих современных систем альтернативы уже нет. Скажите, вы часто слышите о проектах создания систем с нуля — в банке, страховой компании, индустрии развлечений, где не требуются высокая производительность и масштабируемость? Думаю, нет. Можете себе представить приложение на смартфоне, в котором ответ на ваш запрос приходит через час? Однако 20 лет назад все приложения были такими. Но мы забываем, что за нажатием кнопки запуска приложения на телефоне стоит обработка огромных объемов данных, которые постоянно растут. Поэтому и необходима обработка именно в памяти, то есть та, которая максимально приближена к оборудованию, иначе добиться ускорения уже физически невозможно. Сегодня нет технологий и подходов, которые бы обеспечивали более высокую скорость и большую масштабируемость, чем обработка и хранение в памяти.

Однако все это не так просто, как кажется, хотя бы по причине разнообразия задач и данных. Пять-семь лет назад, когда мы только поднимали наш бизнес, слышали множество выражений: зачем нам такие технологии, если мы не занимаемся запуском ракет или исследованием залежей углеводородов? Но за последние годы ситуация изменилась, и теперь к нам приходят стартапы, которые не знают, как справиться с ежедневной обработкой 400 Тбайт данных.

**Что сдерживает распространение технологий обработки в памяти и где прежде всего востребованы ваши решения?** Циклы развития ИТ-индустрии сокращаются, и если 20 лет назад такой цикл составлял 10–15 лет, то сегодня компании каждые два-три года обновляют свои программные стеки. Системы *in-memory computing* пока не стали стандартом для всех организаций во всем мире, но есть определенные категории предприятий, которые уже не могут без них обойтись. Например, на Уолл-стрит это абсолютно необходимая технология, и больше половины наших клиентов — это финтех-компании, для которых скорость обработки данных находит прямое отражение в деньгах. Они первыми обратились к технологии обработки в памяти еще в 1990-е и с тех пор являются локомотивом ее продвижения. Но у нас много других интересных клиен-

тов, например, в области биоинформатики, где ведется создание новых лекарств, в разработке мобильных приложений и др. Появляются клиенты, о заинтересованности которых в наших технологиях мы раньше даже и подумать не могли. Недавно к нам обратились заказчики из Autodesk, пожелавшие добавить к своей системе проектирования интересную возможность — кнопку, нажав на которую, пользователь отправляет чертеж детали в производственные студии по всему миру, имеющие 3D-принтеры. Конструктору выставляется предложение по цене, а система автоматически выбирает производство с оптимальными затратами. Когда в Autodesk была запущена эта функция, то все системы компании сразу «упали», поскольку не могли справиться с таким огромным числом студий, желающих предложить свои услуги и конкурирующих за право изготовить изделие по чертежу.

И таких примеров все больше: сегодня очень трудно найти бизнес, в котором скорость и масштабируемость обработки данных были бы неважны. Меняются бизнес-процессы, меняются и ожидания клиентов, как розничных, так и корпоративных.

### **А какова ситуация в России?**

Мы пытались продвигать здесь наши технологии сразу после кризиса 2008 года, вели переговоры с ФНС, «Газпромом», с другими крупными компаниями, которые в один голос заявляли, что эти «космические» технологии им не нужны. А через четыре года мы начали сотрудничать со Сбербанком и сейчас уже работаем с рядом российских компаний, с ФНС и др. В России поняли, что если требуется конкурировать наравне со всем миром, то такие системы нужны.

**Ваш продукт — это своего рода «прослойка» между памятью и СУБД. Но есть мнение, что современные СУБД не готовы к работе даже с технологиями флеш-памяти, что уж говорить об *in-memory computing*.**

Если посмотреть на технологии *in-memory* двадцатилетней давности, то это было именно так: системы выполняли только роль кэша между базой данных и приложением. Но за последнее время сложность подобных решений значительно выросла, и идея «прослойки» уходит. Сегодня GridGain сама по себе может полностью заменить СУБД, поскольку поддерживает все те же ключевые функции: SQL, транзакции, высокую доступность и др. Более того, мы предоставляем в разы больше функциональности, чем традиционные

СУБД: потоковую репликацию, вычислительный грид (традиционные высокопроизводительные системы), интеграцию со Spark и Hadoop, файловую систему в памяти и многое другое.

Именно поэтому нас выбрали Сбербанк и Barclay's, ведь GridGain — не просто инструмент вычисления в памяти, а возможность заменить все базы данных, которые сегодня развернуты в финансовых учреждениях. В Сбербанке это и происходит — одна из идей состоит в том, чтобы перейти с СУБД Oracle на GridGain. Базы данных в современном мире становятся местом, где складируются данные на долговременное хранение, размещаются резервные копии, но это обходится компаниям слишком дорого. Зачем Сбербанку ежегодно платить десятки миллионов долларов в год лицензионных отчислений за резервную систему хранения, когда вся обработка будет на GridGain? Для этих целей гораздо проще развернуть, например, Hadoop.

Сама идея СУБД начинает постепенно вымываться, но, естественно, этот процесс займет определенное время.

### **Насколько сложен процесс миграции с Oracle на GridGain?**

Сложность перехода определяется конкретным проектом. Имеются проекты, где на GridGain переводятся уже существующие системы, а есть такие, где все делается с нуля с использованием наших технологий. В Сбербанке уже введены в продуктивную эксплуатацию первые системы на базе GridGain, причем меньше чем через год после подписания контракта, что для такого масштабного банка поразительный результат. Мы видим, как работают другие, более консервативные банки — например, американские, для которых любое небольшое изменение в бизнес-процессе занимает годы.

Со временем Сбербанк станет брендом, объединяющим разные бизнесы, и его руководство стремится подготовить ИТ-экосистему к такой трансформации. Реализовать эти амбициозные планы на фундаменте традиционных реляционных баз данных, работающих с диском, невозможно.

### **Какие еще у вас есть клиенты с похожими по масштабу задачами?**

Мы работаем с Barclay's, и с Citibank, но аналогов тому, чего стремится достичь Сбербанк, пока нет. И поскольку у нас команда преимущественно российская, нам очень приятно, что банк именно из России

настолько инновационен в самом правильном, непафосном смысле этого слова. По мнению аналитиков Gartner, в области in-memory computing Сбербанк является банком номер один в мире.

## Что сегодня представляет собой рынок in-memory computing, кто ваши конкуренты?

Из больших компаний — это SAP с системой HANA, хотя опции обработки в памяти имеются в продуктах Oracle, Microsoft и IBM. Но всем этим компаниям надо обслуживать свои традиционные СУБД — эту громадную «дойную корову», которой они добавляют «бантики» работы в памяти, но принципиально сделать с ней ничего не могут, так как это важнейший источник дохода. Это монстры со своим рынком, и мы с ними практически не конкурируем.

Есть интересные стартапы вроде VoltDB и MemSQL. Они реализуют традиционные базы данных SQL, в которых диск заменен на память. Есть ряд компаний, подход которых аналогичен нашему, а мы представляем больше, чем просто СУБД: наша система реализует целый комплекс функций, формирующих полноценную структуру работы с данными, потому она и называется In-Memory Data Fabric. Среди компаний, придерживающихся схожего подхода, можно назвать Hazelcast, которая работает здесь в Кремниевой долине, Gigaspace из Израиля, и, наверно, на сегодня все. Поле конкуренции не такое большое. Нам помогает то, что мы развиваем свою систему в рамках модели Open Source и лидируем в этой области. Вообще, модель Open Source очень важна в сфере инфраструктурного ПО; в противном случае, думаю, у вас практически нет шансов. Это подтверждает успех и Hadoop, и Spark, и компании, которые развивают свои решения на основе этих технологий.

## Для Hadoop предлагается много коммерческих реализаций. С Ignite происходит что-то подобное?

Начавшийся было рост числа компаний вокруг Hadoop быстро прекратился, все поняли, что достаточно двух-трех дистрибутивов, а четыре-пять уже перебор. Поэтому сейчас здесь по сути осталось три серьезных игрока: Cloudera, Hortonworks и MapR, — а остальные сошли с дистанции, даже Intel закрыла полумиллиардный проект по Hadoop. Ничего, кроме головной боли для пользователей, большое число коммерческих версий проекта с открытым кодом не приносит. Если посмотреть на Spark, то как была одна компания, кото-

рая занималась его развитием (Databricks), так она и осталась, хотя никому ничто не мешало найти инвестиции и начать что-то делать вокруг Spark. Само сообщество пользователей решило, что это не нужно. Действительно, гораздо проще сконцентрироваться на разработке самого проекта с открытым кодом и иметь одну, максимум две компании, которые реализуют для него корпоративные версии.

Проблема в том, что чем больше коммерческих компаний вовлечены в проект, тем больше различных трений, конфликтов и тем сложнее проекту двигаться вперед. Hadoop прошел через все это, и одна из причин, почему он так медленно развивался, демонстрируя настолько низкое качество, состоит в том, что проектом занималось множество компаний, каждая из которых тянула одеяло на себя.

Для Apache Ignite пока есть только одна компания, разрабатывающая корпоративную версию, — это GridGain. Мы предоставляем стабильную версию с дополнительными функциями и стандартным набором сервисов поддержки, консалтинга и т. д. Может быть, появятся и еще компании, а может быть, и нет, но, даже если GridGain завтра перестанет существовать, проект Ignite продолжит развиваться, и это одно из преимуществ модели Open Source. Когда мы приходим в крупную организацию, возникает вопрос: как можно смотреть в будущее, работая сегодня с небольшим стартапом в 100 человек? Но риски снижает тот факт, что за нами стоит мощный проект с открытым кодом.

## Компания GridGain до сих пор сохраняет статус стартапа. Когда она станет зрелым бизнесом?

Согласно известному определению, стартап — это временная организация, которая находится в поиске устойчивой бизнес-модели. Абсолютное большинство компаний в Кремниевой долине — это стартапы, независимо от того, сколько человек в них работает. Думаю, мы останемся стартапом еще долго, находясь в поиске устойчивой бизнес-модели, потому что меняются технологии, меняются ожидания, сокращаются циклы развития технологий. Например, на изменение бизнес-модели очень сильно повлияли облака, что сильно ударило по таким компаниям, как Microsoft, Oracle и др. С появлением и взрослением подобных технологий будет меняться и наша бизнес-модель. Я вообще приверженец идеи, что практически любой бизнес, если он правильно управляется, всегда остается стартапом, потому

что постоянно находится в поисках этой модели. Внешние факторы все время меняются, и если 100 лет назад можно было основать IBM и следующие 50 лет развиваться в рамках одной и той же бизнес-модели, то больше такого уже не будет.

Конечно, мы будем расти, последние три года GridGain ежегодно удваивает продажи, и на следующий год у нас тоже большие планы. Мы постоянно нанимаем людей по всему миру — в Европе, в России, в Америке, но по сути своего бизнеса оставляемся стартапом.

## Ваши разработчики все из России? Какие к ним предъявляются требования?

В нашем штате около 70% разработчиков из России. Я сам был воспитан с той мыслью, что наши специалисты — лучшие в мире, однако это миф. Безусловно, в стране приличное базовое образование, но нашим выпускникам кардинально не хватает реального опыта участия в серьезных проектах. Например, в любом крупном центре США молодые специалисты очень быстро накапливают практический опыт, работая либо в Google, либо в Facebook, Twitter или Microsoft, хорошо понимая, что такая современная коммерческая разработка программного обеспечения. А в России этого просто нет, и это главная проблема для нас.

## Каковы планы по развитию технологии GridGain?

Мы не станем отклоняться от стратегической линии и делать принципиально новые продукты, но проекту предстоит глубокое развитие в силу того, что меняются требования к нему. Для крупных организаций сегодня все более актуальны задачи цифровой трансформации, и с этим связаны ключевые направления совершенствования технологии GridGain. Одно из наиболее интересных и важных направлений на этот год — машинное обучение, другая область — HTAP (hybrid transactional analytical processing).

В ИТ исторически сложилась полная разобщенность транзакционной и аналитической обработки данных — это всегда были два абсолютно разных мира, что увеличивало стоимость решений. А сейчас наметилась тенденция перехода к общей, гибридной системе, объединяющей обработку транзакционных и аналитических данных. Мы хотим стать пионером в области HTAP.

**Наталья Дубова (osmag@osp.ru) — научный редактор, «Открытые системы. СУБД» (Москва).**

# Кросс-языковая идентификация авторов публикаций

Идентификация авторов публикаций важна для определения их научного рейтинга, однако при обработке имен русскоязычных авторов в англоязычных публикациях нередки ошибки, приводящие к некорректным вычислениям.

*Ключевые слова:* идентификация авторов, наукометрия, РИНЦ  
*Keywords:* identity resolution, scientometrics, Scopus

Зинаида Апанович

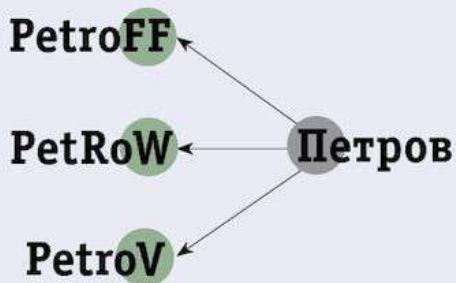
Сегодня большое распространение получили такие крупномасштабные базы знаний, как Google Knowledge Vault, Deep Dive, Microsoft Academic Graph и др., использующие автоматические методы интеграции данных из множества источников и извлечения фактов из текстов, что, однако, является причиной ошибок, связанных с различиями в схемах источников данных, и ошибок идентификации таких сущностей, как авторы публикаций. Для устранения этих ошибок разрабатываются методы слияния данных, направленные на обнаружение и устранение ошибок и конфликтов, проверку корректности извлекаемых фактов, оценку надежности источников данных и методов извлечения информации из них. Вместе с тем значительная часть текстов, в частности научных публикаций, создаются не англоязычными авторами, а также переводятся с различных иностранных языков, что усложняет задачу интеграции множественных источников данных, порождая проблему кросс-языковой идентификации именованных сущностей и, в частности, задачу кросс-языковой идентификации авторов научных публикаций.

Точное установление авторов научных публикаций — важнейший фактор определения рейтинга ученого или научного сотрудника, что делает актуальной задачу кросс-языковой идентификации при интеграции или сравнении разнозычных баз данных, а также при пополнении любой

научной базы знаний информацией об англоязычных публикациях русскоязычных авторов. Однако эксперименты по сопоставлению контента разнозычных баз знаний [1, 2] показали, что такие базы, как библиографическая база данных по информатике DBLP или приложение RKBExplorer.com, интегрирующее информацию об исследователях, публикациях и научных организациях из большого количества разнородных источников, и ряд других изобилуют ошибками при установлении авторов. Например, публикации нескольких разных авторов идентифицировались как принадлежащие одному и наоборот, что ведет кискажению показателей научной продуктивности, основанных на учете цитирования работ.

Надо отметить, что идентификация авторов «в рамках» одного языка проводится вполне успешно, тогда как задача кросс-языковой идентификации сравнительно нова и решается либо путем организационных мер, либо алгоритмически. Примером организационного подхода может служить некоммерческий проект ORCID, позволяющий каждому желающему (автору или организации) получить свой уникальный идентификатор, а затем вручную связать с ним все возможные способы написания имени, публикации и места работы. Имеется также сервис VIAF (Виртуальный международный авторитетный файл, viaf.org), собирающий информацию о принятых в разных странах формах написания имен.

На алгоритмическом уровне идентификация сущностей может проводиться



путем сравнения атрибутов в контексте открытых связанных данных, например, при помощи программного инструментария SILK. Отдельную большую группу составляют методы на базе эвристик, использующих информацию о соавторах и о месте публикации материала (название конференции, журнала). Однако все это не избавляет от ошибок, причиной которых чаще всего является неполнота данных, поэтому требуются специальные методы анализа текстов.

Сегодня применяются различные методы установления авторства: анализ на уровне пунктуации, орфографии, синтаксиса, а также анализ лексико-фразеологических и стилистических особенностей. Однако при сравнении англоязычных текстов авторов с русскоязычными именами эти методы не подходят хотя бы причине того, что разные тексты одного и того же автора, скорее всего, переводили разные переводчики. При установлении идентичности сущностей необходимо принимать во внимание особенности построения перевода или транслитерации имен собственных с учетом специфики того или иного языка — требуется комбинированный подход к кросс-языковой идентификации, сочетающий сравнение атрибутов публикаций, текстов публикаций и транслитераций имен авторов.

Во многих англоязычных ресурсах обычно не уделяется должного внимания различным вариантам написания иностранных имен, полученным при помощи транслитерации, поскольку каждый иностранный

# ИТ-университеты

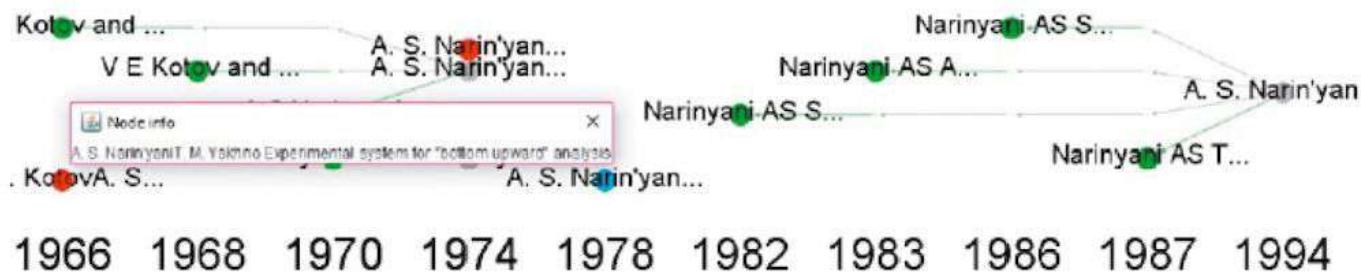


Рис. 1. Фрагмент графа публикаций А. С. Нариньяни, извлеченного из ресурса SpringerLink

язык имеет свои специфические особенности, которые трудно учесть не носителям языка. Что касается русского, то существующие программы генерации англоязычной транслитерации не совсем пригодны для решения задачи идентификации, поскольку генерируют обычно «стандартный» вариант транслитерации, в то время как русскоязычные авторы часто используют в своих публикациях несколько разных нестандартных вариантов написания своего имени. Например, Александр Семенович Нариньани в разных публикациях указывал себя как Nariniani, Narinyani, Narin'yan и Nariniany. Кроме того, в разных публикациях могут использоваться разные варианты транслитерации и сокращения имени и отчества: Alexander S. Narinyani и A.S. Nariniani и т. д. По этой причине для генерации английских транслитераций русскоязычных имен следует использовать разные программы: например, переводчик [google.translate.com](http://google.translate.com) и оригинальную отечественную программу расширенной транслитерации, которая не только генерирует большее количество возможных транслитераций, но и позволяет накапливать обнаруженные нестандартные варианты.

В ИСИ СО РАН создана собственная система идентификации авторов, способная делать это достаточно точно. По русскоязычному имени автора система генерирует все возможные варианты англоязычного написания его имени, а затем по каждому ищет статьи в библиотеке SpringerLink и извлекает оттуда название статьи, места работы автора (если указаны), список всех цитируемых публикаций, аннотацию публикации и полный текст (если имеется). Результат визуализируется в виде графа публикаций, в котором вершины изображают публикации, а ребра — ссылки на другие публикации (рис. 1).

Электронная библиотека SpringerLink ([link.springer.com](http://link.springer.com)), которая, в отличие от специализированных, является библиотекой широкого профиля, содержит полные тексты большинства статей в формате PDF, а если тексты недоступны, то подробную

квазиструктурированную информацию об издании, месте работы авторов (если, конечно, они были указаны в статье), списки цитирований и др. Кроме того, каталог данной библиотеки является одним из источников, используемых крупнейшей в мире библиотечной разноязычной сетью WorldCat.org, что позволяет дополнительно сопоставлять данные.

Если сведения об авторе имеются в Открытом архиве СО РАН [3], то информацию о всех местах его работы можно найти на сайте этого архива. При помощи переводчика Google осуществляется перевод рускоязычного названия организации на английский. Для каждой найденной статьи в библиотеке SpringerLink извлекается место работы запрашиваемого автора и осуществляется нечеткое сравнение с местами работы, полученными для данной персоны на предыдущем шаге в архиве СО РАН. Названия организаций имеют достаточно сложную структуру, и вариантов их написания множество, не говоря уже о вариантах сокращений. В некоторых статьях место работы не указывается вообще или указывается частично (например, РАН). Сравнение в этом случае производится на основе модифицированной версии алгоритма Джаро — Винклера, определяющего меру сходства двух строк и считающегося наилучшим для сравнения коротких строк, таких как имена персон. Далее дата публикации статьи сравнивается со временем работы сотрудника в указанной организации (если таковая информация имеется), а все найденные статьи разбиваются на группы в соответствии с идентифицированным местом работы. Статьи, для которых место работы автора не указано, сравниваются со всеми статьями, размещенными по другим группам, и если текстовое сходство рассматриваемой статьи с публикациями одной из идентифицированных групп превышает некоторое пороговое значение, то статья помещается в эту группу. Сейчас для сравнения текстового сходства имеются две возможности: метод tf-idf и косинусная метрика близости, а также метод LDA

(латентное размещение Дирихле) [4]. Для статей, текст которых оказался неподходящим на тексты ни одной из уже существующих групп, создается новая группа. Для каждой группы создается закладка, названная по одному из известных мест работы заданного автора, и строится граф сходства между статьями, попавшими в каждую группу. Вершина графа — это документ, а ее номер соответствует номеру документа в коллекции. Каждая пара документов в коллекции связана ребром, чей вес ( $W$ ) соответствует сходству между двумя документами. Если величина сходства между двумя документами не превышает установленного порога, то ребро между этими вершинами не создается. В полученном графе похожие документы располагаются ближе друг к другу.

На рис. 1 показан фрагмент ориентированного графа публикаций А. С. Нариньани, извлеченного из набора данных SpringerLink. Каждая вершина графа изображает одну публикацию. Все публикации упорядочены горизонтально по годам. Ребро между двумя публикациями направлено справа налево и означает, что правая публикация цитирует в своем списке литературы левую. Поскольку названия публикаций достаточно длинные, на экран выдаются только первые 15 символов из названия, но всегда можно увидеть более подробную информацию о публикации, как это показано на рис. 1 для работы A.S. Narin'yan, T.M. Yakhno «Experimental system for bottom upward analysis», опубликованной в 1978 году. Примечательно, что только две из двенадцати публикаций, обнаруженных на SpringerLink, присутствуют в списке публикаций Нариньани, размещенных в elibrary.ru, а полные англоязычные тексты SpringerLink обнаружены для семи его публикаций. Зеленым цветом выделены публикации Нариньани, на которые есть ссылки в англоязычных источниках, но их тексты отсутствуют в SpringerLink. При этом были обнаружены такие варианты написания фамилии, как Nariniani и Narin'yan с различными вариантами сокращений имени и фамилии. Надо отметить, что на

таком библиографическом сайте, как DBLP, имеется три непересекающихся множества публикаций Нариньяни, которые распределены между тремя совершенно разными персонами с разными вариантами написания имени и фамилии (A. Narinyani, A. S. Narinyani, Alexander S. Narin'yani). В то же время на сайте SCOPUS имеется другой набор публикаций этого же автора, и там его фамилия уже пишется как Narin'yani или Narin'Yani. Таким образом, на четырех ресурсах оказались пересекающиеся, но не совпадающие множества публикаций.

Что касается данных в библиотеке SpringerLink в целом, то следует отметить значительный разброс в объеме доступной информации о публикациях (от пары абзацев до нескольких десятков страниц), что существенно влияло на точность идентификации. К тому же результаты проверки программы на тестовой выборке из 100 персон (около 3000 публикаций) показали, что примерно в 80% случаев в публикациях не было информации о полном имени персоны, имелись только инициалы. Место работы персон было указано примерно в 70% случаев.

На рис. 2 показан пример работы системы [5] при поиске в библиотеке SpringerLink публикаций Валерия Александровича Непомнящего из ИСИ СО РАН. Различные англоязычные варианты написания этого имени показаны в верхней вкладке слева. В средней вкладке слева показаны англоязычные варианты места работы данной персоны. В центре показан граф, изображающий публикации, приписанные Непомнящему из ИСИ СО РАН. Всего в SpringerLink.com было найдено 49 публикаций. Из них автору с написанием имени Valery Nepomniaschy соответственно 3 статьи, V.A. Nepomnyashchii — 1 статья, V.A. Nepomniaschy — 24 статьи, Valery A. Nepomniaschy — 3 статьи, V. A. Nepomnyashchii — 15 статей, V.A. Nepomnyashchii — 3 статьи. Все указанные варианты написания имени, отчества и фамилии использовались В. А. Непомнящим из ИСИ СО РАН, но, кроме него, были обнаружены еще две персоны, одна из которых использовала для своего имени вариант написания V. A. Nepomnyashchii, а вторая — V. A. Nepomnyashchii. На рис. 2 можно видеть, что программа создала две новые группы публикаций для этих персон. В то же время на сайте Scopus было показано пять разных персон с разными вариантами написания фамилии Непомнящий и разными идентификаторами. При этом публикациям реальной персоны В. А. Непомнящего из ИСИ СО РАН соответствовали публикации

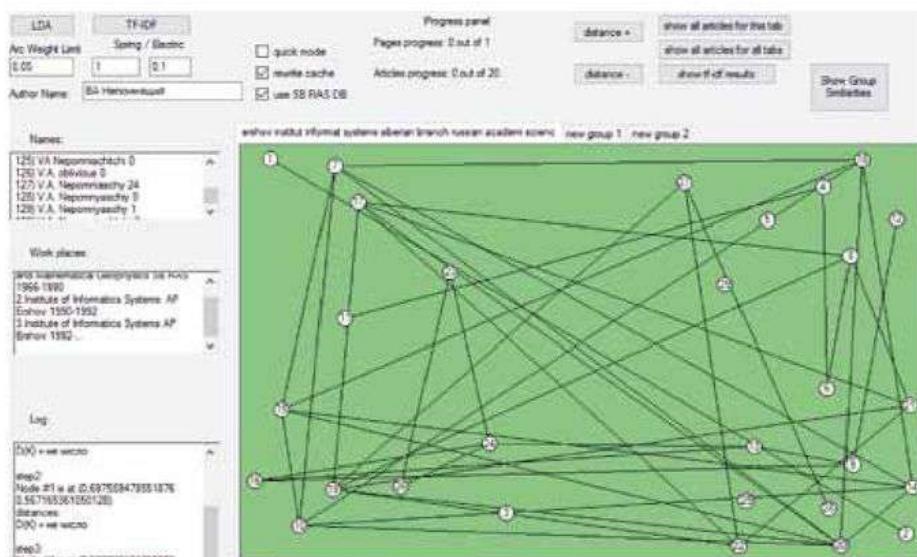


Рис. 2. Публикации, идентифицированные как принадлежащие В.А. Непомнящему из ИСИ СО РАН

четырех «виртуальных» персон, имеющих в Scopus разные идентификаторы: персона Nepomniaschy с идентификатором Scopus 6603218491, персона V. A. Nepomnyashchii с идентификатором Scopus 6701781364, персона V.A. Nepomnyashchii без идентификатора с местом работы НГУ и частично персона Nepomnyashchii V.A. с идентификатором Scopus 24076960300.

Таким образом, в системе Scopus за- ведомо ошибочно были «распределены» публикации трех реальных людей, публикации которых присутствуют на сайте, по пяти «виртуальным» персонам, а также произошло объединение в одну группу публикаций Валерия Непомнящего из ИСИ СО РАН и Владимира Непомнящего из Москвы. И таких примеров некорректной работы как Scopus, так и eLibrary обнаружилось множество. Эксперименты также показали, что elibrary.ru может быть полезной при идентификации ныне живущих авторов, не менявших место работы, однако при изучении публикаций персон, которые по разным причинам завершили свою деятельность или сменили место работы, обнаруживается неполнота данных.

\*\*\*

Сегодня ни одна из существующих библиотек не предоставляет полных данных по авторам — сведения о публикациях одного и того же человека могут быть рассредоточены по различным англо- и русскоязычным ресурсам. Поэтому для получения достоверной картины требуется объединение информации из различных источников. Предлагаемая система позволяет решить задачу кросс-языковой идентификации авторов в условиях расширенной трансляции, однако и ей еще требуется разви-

тие в сторону адаптации к сопоставлению произвольной пары русскоязычных и англоязычных источников данных. В итоге это позволит точнее вычислять различные рейтинги активности ученых и более адекватно оценивать их вклад в развитие конкретной области знания. ■

## ЛИТЕРАТУРА

1. Apanovich Z.V., Marchuk A.G. Experiments on using the LOD cloud datasets to enrich the content of a scientific knowledge base // KESW 2013, CCIS 394. — Springer Verlag Berlin Heidelberg 2013. — P. 1–14.
2. Zinaida Apanovich, Alexander Marchuk. Experiments on Russian-English identity resolution//Proceedings of the ICADL-2015 Conference Seoul, South Korea, LNCS 9469. — Springer International Publishing Switzerland 2015. — P. 12–21.
3. Марчук А.Г., Марчук П.А. Особенности построения цифровых библиотек со связанным контентом // Труды RCDL'2010. — Казань, 2010. — С. 19–23.
4. Blei D. M., Ng A., Jordan M. Latent Dirichlet Allocation // Journal of Machine Learning Research. — 2003, № 3. — P. 993–1022.
5. Апанович З.В. Сопоставление данных разнозычных ресурсов и кросс-языковая идентификация авторов // Научный сервис в сети Интернет: труды XVIII Всероссийской научной конференции (19–24 сентября 2016 г., Новороссийск). — М.: ИПМ им. М.В. Келдыша, 2016. — С. 36–45. URL: <http://keldysh.ru/abraw/2016/proc.pdf> (дата обращения: 18.01.2017).

Зинаида Апанович ([apanovich@iis.nsk.su](mailto:apanovich@iis.nsk.su)) — старший научный сотрудник, ИСИ СО РАН (Новосибирск).

# Полвека на переднем крае ИТ

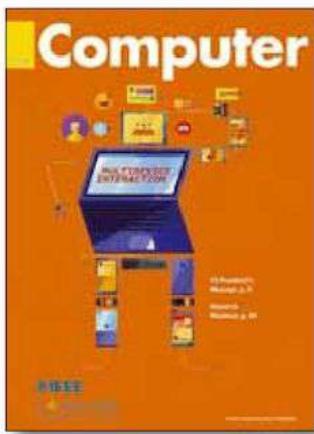
Темы декабря 2016 года, январского и февральского номеров 2017 года журнала Computer (IEEE Computer Society, Vol. 49, No. 12, 2016 и Vol. 50, No. 1, 2, 2017) — взаимодействие компьютерных устройств, будущее мира ИТ и технологии, дополняющие возможности человека.

*Ключевые слова:* расширение возможностей человека, управление распределенными инфраструктурами, человеко-машинный интерфейс

*Keywords:* human augmentation, human-machine interface, management of distributed infrastructures

Александр Тыренко

**В** этом году журнал Computer отмечает полувековой юбилей. Издание общества IEEE Computer Society, впервые вышедшее в 1967 году, вначале носило название Computer Group News и выпускалось каждые два месяца, а спустя три года получило свое нынешнее имя. Публикации журнала ориентированы на участников научно-исследовательского сообщества, студентов и специалистов-практиков в сфере ИТ. Его статьи, по стилю балансирующие между научными докладами и отраслевыми публикациями, неизменно направлены на освещение новых горизонтов компьютерных технологий.



Декабрьский номер журнала Computer посвящен групповому взаимодействию устройств.

С Интернетом сегодня соединено беспрецедентное множество компьютерных устройств разнообразных форм и размеров, имеется и интерес к использованию их совокупных возможностей. Если удастся решить проблемы, связанные

с разнородностью данных, отсутствием стандартов и интероперабельности, можно будет прийти к совершенно новой вычислительной парадигме.

В статье «Энергоэффективное распознавание речи с помощью группы устройств» (Collaborative and Energy-Efficient Speech Monitoring on Smart Devices) Ярно Леппанен (Jarno Leppanen), Микко Пелконен (Mikko Pelkonen), Хайлэн Го (Haipeng Guo), Самули Хемминки (Samuli Hemminki), Петтери Нурми (Petteri Nurmi) и Сасу Таркома (Sasu Tarkoma) описывают систему распознавания речи, которая автоматически выбирает лучший, активный источник звука в помещении путем поочередного включения микрофонов на смартфонах, находящихся вблизи говорящего. При этом имеется возможность балансировки качества звука и управления затратами электроэнергии в системе.

В работе «Программирование коллективного поведения роботов в распределенных сетях» (Swarm-Oriented Programming of Distributed Robot Networks) Карло Пинчироли (Carlo Pinciroli) и Джованни Белтраме (Giovanni Beltrame) разъясняют, как средствами языка программирования Buzz можно обеспечить совместное выполнение общей задачи большой группой роботов, выбрав при этом области специализации для индивидуальных участников.

В статье «Организация взаимодействия межплатформенного контента для удержания внимания зрителя» (Interdevice Media: Choreographing Content to Maximize Viewer Engagement) Тимоти Нит (Timothy Neate), Мэтт Джонс (Matt Jones) и Майкл Эванс (Michael Evans) анализируют возможности совмещения просмотра телевизора и одновременного взаимодействия с мобильными устройствами. Рассматриваются способы создания перекликающегося контента для телевизора и смартфона, а также вопросы оценки эффективности различных методов управления зрительским вниманием.

Томас Бурес (Tomas Bures), Франтишек Пласил (Frantisek Plasil), Михал Кит (Michal Kit), Петр Тума (Petr Tuma) и Никлас Хох (Nicklas Hoch) в статье «Программные абстракции для взаимодействия компонентов в Интернете вещей» (Software Abstractions for Component Interaction in the Internet of Things) рассказывают о парадигме проектирования программных систем, основанной на высокогорневых абстракциях, позволяющих описывать экосистемы Интернета вещей, для которых характерны динамизм архитектуры, расширяемость и способность к автоматической адаптации.

Александр Котт (Alexander Kott), Анантрам Свами (Ananthram Swami) и Брюс Вест (Bruce West) написали статью «Интернет боевых вещей» (The Internet of Battle Things), в которой обсуждают превращение полей сражения будущего в сети Интернета вещей, требующие сложнейших механизмов управления, мониторинга и защиты.

Публикации январского номера посвящены будущему мира ИТ, в частности проблеме возвращения отрасли на путь масштабируемости. Кроме того, обсуждаются инновации в области человеко-машинного взаимодействия, а именно: принципы дизайна интерфейсов, основанные на моделировании действий пользователя, применение различных технологий для распознавания активности человека.



В статье «Перезагрузка ИТ: дальнейшие перспективы» (Rebooting Computing: The Road Ahead) Томас Конте (Thomas Conte), Эрик Дебенедиктис (Erik DeBenedictis), Паоло Гаргини (Paolo Gargini) и Эли Трек (Elie Track) рассказывают о развитии инициативы IEEE Rebooting Computing, направленной на переосмысление всех аспектов цифровой трансформации, от особенностей оборудования до пользовательского интерфейса. Одна из задач инициативы — обход физических ограничений, угрожающих дальнейшему действию закона Мура в эпоху активного развития систем машинного обучения и анализа Больших Данных, сопровождаемых беспрецедентным ростом потребностей в вычислительных мощностях. Авторы напоминают, что с изменением основополагающих аппаратных технологий придется менять и архитектуру ПО для адаптации к новым решениям.

Статья «Начало развития периферийных вычислений» (The Emergence of Edge Computing), которую представил Махадев Сатьянараянан (Mahadev Satyanarayanan), освещает перспективы применения вычислительных архитектур, объединяющих Интернет вещей, мобильные устройства и облачные сервисы в «IoT-облако». Основная вычислительная нагрузка в такой архитектуре приходится на облачные серверы, но если временно размещать облачные ресурсы на периферийных компьютерах, находящихся через один-два транзитных узла от клиентского мобильного приложения, то можно существенно улучшить «отзывчивость» сервисов. В числе других преимуществ — возможность маскировки временных сбоев облака и эффективный контроль выполнения политик приватности.

Анти Оуласвирта (Antti Oulasvirta) в публикации «Дизайн пользовательских интерфейсов с комбинаторной оптимизацией» (User Interface Design with Combinatorial Optimization) описывает

методику конструирования пользовательских интерфейсов с применением прогнозных моделей человеческого восприятия и поведения. В статье предлагается идея инструментария упрощенной разработки пользовательских интерфейсов, который в процессе работы сам подсказывает дизайнеру идеи.

Хао Ван (Hao Wang), Дань У (Dan Wu) и Дацин Чжан (Daqing Zhang) опубликовали статью «Распознавание активности человека с помощью сигналов Wi-Fi» (Toward Centimeter-Scale Human Activity Sensing with Wi-Fi Signals), в которой предлагают метод высокоточной идентификации активности человека в режиме реального времени с применением модели распространения радиосигнала, основанной на зонах Френеля. Система, например, распознает жесты и показатели жизнедеятельности, позволяет определить, что пожилой человек упал и не двигается, а также сможет заранее «почувствовать», когда водитель готов уснуть за рулем.



Тема публикаций февральского выпуска журнала Computer — расширение возможностей человека с помощью ИТ, стремительно меняющих мир и людей и компенсирующих недостатки человека. Технологии расширения возможностей индивида (*human augmentation*) предназначены для повышения продуктивности его деятельности, а также усиления или восстановления способностей тела и мозга. Они позволяют улучшить здоровье, качество жизни и функциональные способности.

Флориан Валь (Florian Wahl), Жуй Джан (Rui Zhang), Мартин Фрайнд (Martin Freund) и Оливер Амфт (Oliver Amft) опубликовали статью «Персонализация компьютеризированных очков, изготавливаемых методом 3D-печати» (Personalizing 3D-Printed Smart Eyeglasses to Augment

Daily Life). В ней описывается процесс цифровой адаптации конструкции умных очков к конкретному пользователю. После распечатки на 3D-принтере человек получает оправу со встроенными датчиками, которая будет идеально смотреться и сидеть комфортнее, чем выпускаемые серийно. Среди возможных функций самих очков — анализ ритма сна и бодрствования, контроль питания и т. п.

Статья «Третий глаз: шопинг-ассистент для слабовидящих» (Third Eye: A Shopping Assistant for the Visually Impaired), которую подготовили Питер Зинтара (Peter Zientara), Суин Ли (Sooyeon Lee), Гас Смит (Gus Smith), Рори Бреннер (Rorry Brenner), Лоран Итти (Laurent Itti), Мэри Россон (Mary Rosson), Джон Кэррол (John Carroll), Кевин Ирлик (Kevin Irlick) и Виджайкришнан Нараянан (Vijaykrishnan Narayanan), представляет систему расширения возможностей людей с нарушениями зрения. Она состоит из очков со встроенной камерой и перчаток, вибрацией указывающих человеку нужное направление движения. Система позволяет ориентироваться и находить товары в продуктовом магазине, помогая выбирать необходимое. По словам авторов, испытания описываемых технологий принесли весьма многообещающие результаты.

Чэн Чжан (Cheng Zhang) и его соавторы в статье «Биоакустическая связь через тело человека» (Bioacoustics-Based Human-Body-Mediated Communication) представляют метод передачи информации между устройствами через человеческое тело с помощью акустических сигналов. А в статье «Расширение возможностей мозга с помощью нейротехнологий» (Does Neurotechnology Produce a Better Brain?) Раджан Бхаттхарья (Rajan Bhattacharyya), Брайан Коффман (Brian Coffman), Джейхун Чоу (Jaehoon Choe) и Мэттью Филипс (Matthew Phillips) описывают принципы дополнения когнитивных способностей путем неинвазивной стимуляции головного мозга, применяемой в сочетании с регистрацией сигналов организма. Обсуждается использование нейротехнологий такого рода для целей обучения и обеспечения безопасности, а также для улучшения моторных функций. Кроме того, в статье рассматриваются этические вопросы применения подобных средств. ■

Александр Тыренко (shoorah@osp.ru) — обозреватель «Computerworld Россия» (Москва).

# OPEN SYSTEMS. DBMS

The journal was founded in 1993

IT for Business  
Innovative Technology for Computer Professionals

## Editorial Staff

Dmitry V. Volkov, Editor-in-Chief, Senior Research Fellow, Keldysh Institute of Applied Mathematics

Natalia A. Dubova, Senior Reviewer

## Editorial Board

Valery D. Adzhiev, PhD in Computer Science, Senior Research Lecturer, The National Centre for Computer Animation, Bournemouth University, UK

Fuad T. Aleskerov, DSc in Technical Science, Professor, National Research University Higher School of Economics

Mikhail M. Gorbunov-Possadov, DSc in Physics and Mathematics, Assistant professor, Moscow State University, Keldysh Institute of Applied Mathematics, Russia

Yuri A. Zelenkov, DSc in Technical Science, Professor, Financial University, Russia

Sergey D. Kuznetsov, DSc in Physics and Mathematics, Professor, Moscow State University, Russia

Sergey O. Kuznetsov, DSc in Physics and Mathematics, Professor, National Research University Higher School of Economics, Russia

Mikhail B. Kuzminsky, PhD of chemistry, Senior research fellow, Institute of Organic Chemistry, Russian Academy of Science

Alexander I. Legalov, DSc in Technical Science, Professor, Siberian Federal University

Vladimir A. Suchomlin, DSc in Technical Science, Professor, Moscow State University, Russia

Pavel B. Khratmsova, PhD in Computer Sciences, Assistant professor, National Research Nuclear University MEPhI, Russia

Viktor Z. Shnitman, DSc in Technical Sciences, Professor, Moscow Institute of Physics and Technology, Russia

Igor G. Fiodorov, PhD, Professor, Moscow State University of Economics, Statistics and Informatics (MESI), Russia

Leonid K. Eismont, PhD of physics and mathematics, Science adviser, RSI «KVANT», Russia



Editorial Staff  
Design  
Maria Ryzhova

Cover Design  
Denis Kirkov

Administrative Staff  
President

Mikhail E. Borisov

General Manager  
Galina A. Gerasina

Director, IT Media Group  
Pavel V. Khristov

Commercial Director  
Tatiana N. Filina

**Circulation:** Open Systems Journal (ISSN 1028-7493) is published bi-monthly by the Open Systems Publishing. Open Systems Headquarters, Dobrolyubova pr., 3, bld.3, 127254, Moscow, Russia, 127254; voice +7 495 725-4780/84, +7 499 703-1854; fax +7 495 725-4783.

**Editorial:** Unless otherwise stated, bylined articles, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in Open Systems Journal does not necessarily constitute endorsement by the Open Systems Publishing. All submissions are subject to editing for style, clarity, and space.

Permission to reprint/republish this material for commercial, advertising, or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the OpenSystems Publishing.

Copyright© 2017 Open Systems Publishing. All rights reserved. Issue published under the support of the Federal Agency for Press and Mass Communications.

# 2017, Volume 25, Number 1

## COVER FEATURES

### TRUSTED OPERATING SYSTEMS

#### 12 KasperskyOS: Everything is Forbidden Which is not Allowed

In 2016, Kaspersky Lab announced the availability of its own operating system aimed at securing the operation of network devices by protecting them not only from external threats but also from each other.

Andrey Nikishin ([Andrey.nikishin@kaspersky.com](mailto:Andrey.nikishin@kaspersky.com)), Head of Department for Technology Projects Development, Kaspersky Lab (Moscow).

#### 14 Alt OS: an Enterprise Class Platform

The Alt operating system developed in Russia enables deploying secure scalable platforms to support corporate IT infrastructure lifecycle.

Dmitry Derzhavin ([dd@basealt.ru](mailto:dd@basealt.ru)), lead engineer, Basealt SPO (Moscow).

#### 16 MAKS: A Real-Time Embedded Software Platform

AstroSoft has developed a real time operating system which combines not only basic functions expected from this type of a platform but also a number of differentiating features allowing to speed up embedded software development for devices based on microcontrollers.

Alexander Kucherov ([alexander.kucherov@astrosoft.ru](mailto:alexander.kucherov@astrosoft.ru)), MAKS RTOS Project Manager, AstroSoft (St. Petersburg).

#### 19 Mandatory Access Control in Operating Systems and Databases

While there is no mandatory access control in Linux, its use for various applications requiring privacy protection creates the need to secure the open-source OS working in conjunction with databases. In those settings, ensuring security requires a cross-platform solution.

Valery Popov ([v.popov@postgrespro.ru](mailto:v.popov@postgrespro.ru)), Head of IT Security and Certification Group, Postgres Professional (Moscow).

## PLATFORMS

#### 8 The Chinese CPU and Supercomputer Way

By the end of 2016, Western countries had to recognize that peak performance of both supercomputers and CPUs from China is greater than that of similar U.S. systems, with Chinese engineers having proved they are able to surpass the West in high-performance computing.

Mikhail Kuzminsky ([kus@free.net](mailto:kus@free.net)), research fellow, N.D. Zelinsky Institute of Organic Chemistry (Moscow).

## SECURITY

#### 22 Privacy Risks in Intelligent User Interfaces

Intelligent user interfaces (in games, for example) provide opportunities for producing a high-quality, contextually relevant user experience. However, they also raise the specter of privacy violations. The authors review some of the ways in which user interfaces could glean a user's private information; then the authors highlight the risks therein, and discuss ways of mitigating those risks.

Christopher J. Hazard ([cjhazard@hazardoussoftware.com](mailto:cjhazard@hazardoussoftware.com)), founder of Hazardous Software; Munindar P. Singh ([singh@ncsu.edu](mailto:singh@ncsu.edu)), computer science professor, North Carolina State University.

## SOFTWARE ENGINEERING

#### 25 Building Critical Applications Using Microservices

With system software bugs to be inevitably used for attacks, mission critical applications must not be dependent on the correctness of low-level code. The use of microservices and protected memory areas such as Intel Software Guard Extension minimizes trusted computing base and ensures the necessary application reliability without trade-offs.

Christof Fetzer ([christof.fetzer@tu-dresden.de](mailto:christof.fetzer@tu-dresden.de)), professor of computer science, Technische Universität Dresden.

## INTERNET OF THINGS

#### 28 The Internet of Battle Things

On the battlefields of the future, multitudes of intelligent things will be communicating, acting, and collaborating with one another and with human warfighters. This will demand major advances in science and technology.

Alexander Kott, Ananthram Swami, Bruce J. West, ([alexander.kott1.civ](mailto:alexander.kott1.civ), [ananthram.swami.civ](mailto:ananthram.swami.civ), [bruce.j.west.civ@mail.mil](mailto:bruce.j.west.civ@mail.mil)), research scientists, US Army Research Laboratory.

## EXPERIENCE

#### 31 Open Source Software Adoption Lessons from Linux in Munich

Over 10 years, the city of Munich migrated 15,000 PCs from Windows to the open source Linux operating system. Based on this case, the authors propose recommendations for evaluating and calculating the risks of open source software adoption.

Mario Silic ([mario.silic@unisg.ch](mailto:mario.silic@unisg.ch)), postdoctoral researcher at Institute of Information Management, University of St. Gallen, Switzerland; Andrea Back ([andrea.back@unisg.ch](mailto:andrea.back@unisg.ch)), full professor of management and information systems, University of St. Gallen, Switzerland.

## THE WORLD

#### 34 Teaching programming during IT revolutions

In IT there is a permanent revolution: technologies, tools change, fundamentally new solutions appear. How in these conditions to organize the learning process, to prepare competent programmers, based on fundamental knowledge in the implementation of applied developments?

Victor Ivannikov

#### 40 It All Comes Down to Memory

While offering high performance and scalability not achievable with traditional solutions, the in-memory storage and processing technology developed by GridGain Systems is capable of much more than that due to the diversity of workloads and data it is capable of handling.

Natalya Dubova ([osmag@osp.ru](mailto:osmag@osp.ru)), science editor, Open Systems Journal. DBMS (Moscow).

## OS ACADEMY

#### 43 Cross-Language Identification of research publication authors

Correct identification of authors of research publications is an important factor of evaluating their productivity. Regrettably, errors of identification of Russian authors of English publications are still frequent and lead to an incorrect evaluation of h-index. This paper describes an algorithm establishing a cross-language identity of the authors of research publications. The algorithm is based on a combination of extended transliteration of personal names, attribute-based identity resolution, and text analysis of publications.

Zinaida Apanovich ([apanovich@iis.nsk.su](mailto:apanovich@iis.nsk.su)), senior researcher, IIS SB RAS, Russian Federation (Novosibirsk).

## LIBRARY

#### 46 Computer's 50th anniversary

The December, January and February issues of Computer magazine (IEEE Computer Society, Vol. 49, No. 12, 2016, Vol. 50, No. 1, 2 2017) are focused on group interaction of devices, the future of computing, and human augmentation technologies.

Alexander Tyrenko ([shoorah@osp.ru](mailto:shoorah@osp.ru)), reviewer, Computerworld Russia (Moscow).

ПОДРОБНЕЕ  
[www.osp.ru/iz/bigdata](http://www.osp.ru/iz/bigdata)

# BIG DATA

2017

29  
МАРТА

Москва EVENT-ХОЛЛ  
«ИнфоПространство»  
1-й Зачатьевский переулок, дом 4

Организатор



**BIG DATA 2017** – центральное событие года на российском рынке, посвященное обсуждению опыта, актуальных задач и достижений компаний и организаций, которые строят бизнес, основанный на данных

## Главные темы Форума:

- Данные в основе цифровой трансформации: отраслевой опыт
- От машинного обучения к искусственному интеллекту
- Аналитика о людях: большие данные в маркетинге и управлении персоналом
- Аналитика о вещах и событиях: роль данных в Индустрии 4.0
- Данные на службе государства и общества
- Правовое регулирование высоких технологий
- Инфраструктура больших данных

BIG DATA 2017 – это поиск решения основной задачи, которая стоит сегодня перед руководителями бизнеса и ИТ: как научиться лучше понимать большие данные.

## Премиум партнер

**Hewlett Packard  
Enterprise**

## Генеральные партнеры

**CleverDATA**  
Системы для бизнеса  
  
**ORACLE** Platinum Partner

**DELL EMC**

**Bull**  
altis technologies

**IBS**  
Умный выбор  
менеджерских  
технологий

## Партнеры

**lanti telecom**

**DIS**

**ПОЛИМАТИКА**

**VISOLOGY**

**HITACHI**  
Inspire the Next  
©Hitachi Data Systems

**RCNTEC**

Мы ждём вас! По вопросам участия обращайтесь к Ольге Пуркиной:



+7 (499) 703-1854, +7 (495) 725-4780



[kon@osp.ru](mailto:kon@osp.ru)

Реклама

12+



Источник: Honda

## Мотоцикл Honda не падает даже на самом малом ходу

В Honda разработали технологию сохранения равновесия для мотоциклов, способную автоматически удерживать их в вертикальном положении. По замыслу создателей, их система, находящаяся сейчас на стадии прототипа, должна помочь снизить число ДТП, происходящих с мотоциклами в условиях медленного дорожного движения.

Чтобы удерживать равновесие, мотоцикл, регистрируя перемещения своего центра тяжести, варьирует положение переднего колеса, время от времени совершая почти незаметные движения вперед и назад. При этом, как уверяют в Honda, в системе не используется гироскоп, что делает ее более прогрессивной по сравнению с предыдущими механизмами автоматической балансировки двухколесных транспортных средств. В Honda уверены: оснастить системой уже выпускаемые модели мотоциклов не составит труда.

## В «Теремке» подают блины, напечатанные на 3D-принтере

Сеть ресторанов домашней кухни «Теремок» тестирует 3D-принтеры для приготовления блинов. В компании приурочили внедрение инноваций к празднику Масленицы.

На масленичной неделе любой желающий мог зайти в «Теремок» в Новопушкинском сквере и попробовать создать блин по собственному уникальному рисунку.

3D-печать блинчиков вскоре будет запущена в московском «Теремке» на Грузинском Валу. «Теремок» планирует провести тесты 3D-принтеров в нескольких ресторанах. Если они пройдут успешно, сеть может оснастить такими аппаратами свои заведения не только в России, но и в США.

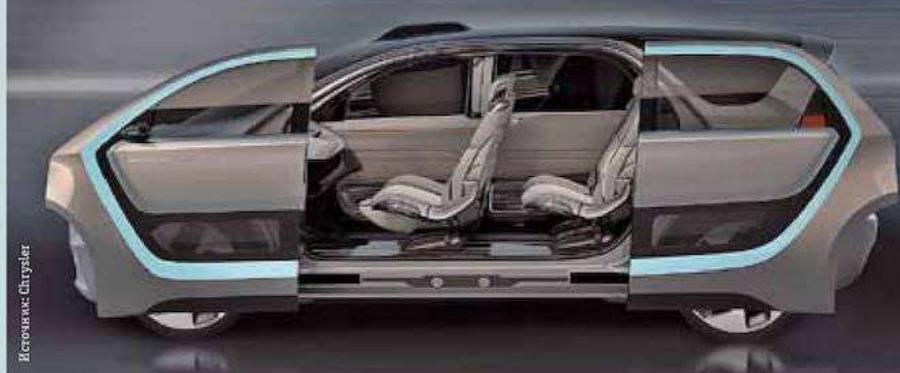


Источник: Continental

## Шины онлайн

Концерн Continental анонсировал выпуск информационно-управляющей системы для грузовых шин ContiConnect. Система отслеживает давление и температуру в шинах через датчики ContiPressureCheck, а затем передает данные на пульт управления. В случае необходимости она предлагает меры по устранению неисправностей. На онлайн-портале появляется полная информация о состоянии шин автомобилей и общей эффективности их работы.

Шины для грузовиков и автобусов Continental теперь снабжаются датчиками, что означает возможность перехода от ручного планового обслуживания к автоматическому мониторингу и целевому сервису.



Источник: Chrysler

## «Цифровой портал» на четырех колесах

Корпорация Chrysler представила Portal — концептуальную модель электромобиля, ориентированного на молодое поколение. Машина снабжена раздвижными дверями, а сиденья в салоне расположены в три ряда, позволяя разместить шесть человек. В Chrysler предпочитают не называть машину минифургоном, хотя прогнозируют, что большинство представителей «цифрового поколения» в следующие десять лет обзаведутся семейством.

Помимо обширной цифровой приборной панели сенсорными экранами оснащены и другие внутренние поверхности Portal. В салоне присутствуют сразу десять доков для зарядки гаджетов, а колонки, как сообщили в Chrysler, способны избирательно направлять звук, так что каждый пассажир сможет слушать свою музыку.

Все водители и пассажиры смогут настраивать собственные профили для Portal, в том числе положение сиденья, руля и т. п. Когда владелец профиля подходит к машине, она автоматически распознает его и устанавливает соответствующие настройки.