Самопалим себе NAS

Джонни Бидвелл демонстрирует подход DIY [Сделай Сам] к производительности при создании сетевого хранилища данных.

По мере удешевления хранилищ данных и разрастания аппетита к данным, все больше людей ищут устройства NAS (network-attached storage — сетевое хранилище данных) для складирования битов своей информации. Предлагаются многочисленные модели готовых устройств от множества производителей и по самым разным ценам. Наиболее распространенными являются устройства с двумя или четырьмя дисками, реализующие все функции, необходимые для домашнего хранилища.

Вас вряд ли удивит, что в сердце многих из этих технологий лежит Linux, поскольку это всего-навсего скромные компьютеры x86 или ARM, снабженные интересными web-интерфейсами и простыми в использовании программами настройки.

В принципе, люди небогатые могут слегка сэкономить, создав собственное устройство NAS. Есть несколько спецдистрибутивов NAS с открытым кодом, и самое популярное трио — тут не обошлось без кровосмешения — *FreeNAS* на базе BSD (переписанный старый проект с тем же названием), *NAS4Free* (развитие первоначального кода *FreeNAS*) и *OpenMediaVault* (проект на базе Debian от автора *FreeNAS*). Все это — замечательные проекты, и они дадут вам всё необходимое: от создания массива до выдачи в доступ ваших пиратских записей Криса де Бурга и прочих файлов. Ну, а вдруг вы хотите настроить что-то самостоятельно? Возможно, вам надо, чтобы ваш компьютер NAS служил еще и медиацентром, или отсылал поток игр *Steam* с компьютера с Windows, или запускал *ownCloud*? Или вы намерены все держать под контролем, особенно в свете нашумевших атак Shellshock на устройства NAS с выходом в Интернет... Короче, давайте выясним, как все это сделать.

> «Вас вряд ли удивит, что в сердце многих из этих технологий лежит Linux.»

Что касается оборудования, важнейшая его часть — жесткие диски. Функции хранилища может выполнять и один большой диск, но лучше приобрести их несколько,

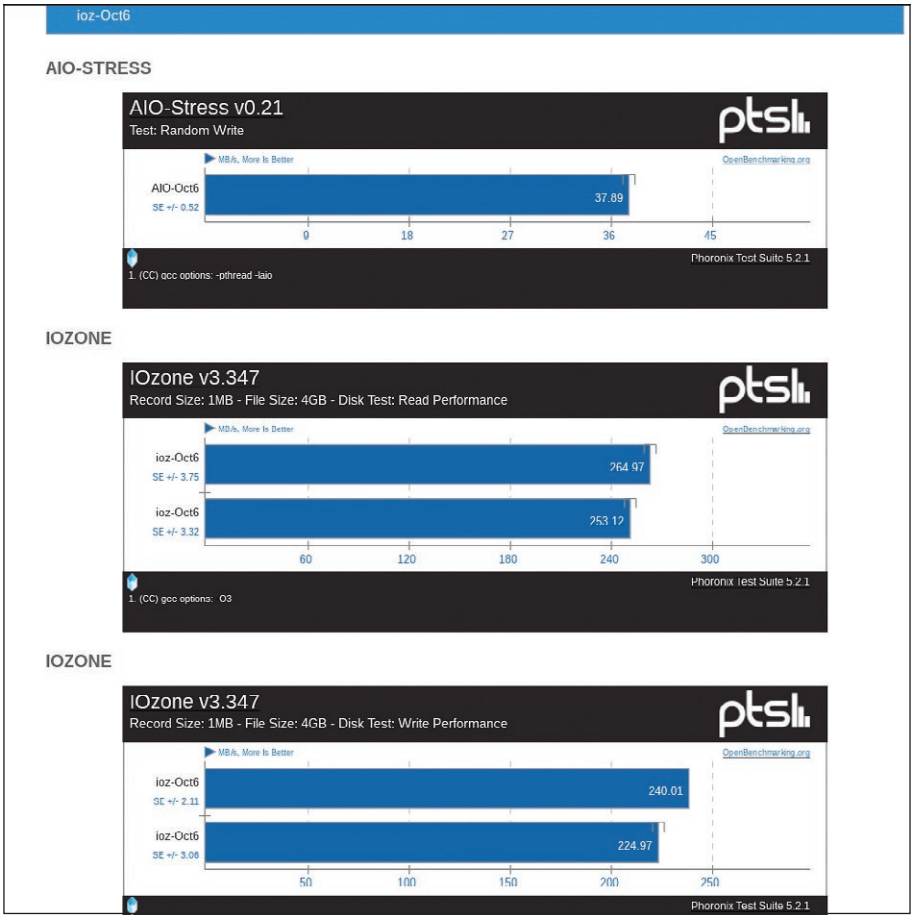
для некоторой избыточности. RAID потребует от двух до шести дисков, и всё будет намного проще и эффективнее, если они будут одного объема. С двумя дисками вы получите конфигурацию RAID1 (где один диск зеркально отражает другой), с тремя дисками у вас будет RAID5 (где данные и блоки четности распределены по дискам для обеспечения повышенной производительности и целостности). Мы выбрали четыре диска, в основном из-за того, что щедрые парни из Western Digital прислали нам четыре диска серии Red по 6 ТБ каждый.

Создание RAID

С четырьмя дисками возможен целый ряд конфигураций RAID, о чем мы расскажем вкратце. Не волнуйтесь — обо всяких тонкостях и экзотических вопросах, связанных с дисками, всё сказано на с. 50. RAID10 — это сочетание уровней 1 и 0, чтобы мы сначала создали массив из двух дисков RAID0 (который не предлагает избыточности, но удваивает производительность), и затем сделали его зеркальное отражение. RAID5 опять же возможен, но не рекомендуется, поскольку при потере одного диска и последующей перестройке массива интенсивный поток операций ввода/вывода заметно увеличит шансы потери еще одного диска — и, следовательно, всех ваших данных. RAID6 предлагает страховку от отказа двух дисков и небольшое увеличение скорости за счет распределения порций данных между дисками, и именно его мы выбрали для нашей структуры. В конечном итоге у нас должно оказаться 12 ТБ полезного пространства и скорость передачи вдвое выше, чем на одном диске.

Можно бы поместить ОС на отдельный раздел одного из ваших дисков RAID, но мы такого не рекомендуем: тогда вам пришлось бы соответственно уменьшить объем всех ваших разделов RAID, ну и вообще неплохо держать такие вещи раздельно. Можно также установить ОС внутри массива, если у загрузчика имеется собственный раздел и ваш образ initrd имеет поддержку *mdadm* (Multiple Disk Administration). И опять же, для хранилища это не подходит.

Мы истратили все внутренние отсеки (и интерфейсы SATA) в нашем небольшом корпусе, поэтому наша ОС отправилась на симпатичный гибридный диск WD Black2 USB3. И это очень неплохо, если только вы случайно не выдернете этот диск во время работы машины. Для простого устройства NAS не нужна полноценная среда рабочего стола, поэтому мы начнем с установки простого Arch Linux. Если вы захотите добавить функции медиа-центра, Arch ничуть не будет против. Об установке Arch на USB-диск можно прочитать на http://bit.ly/ArchOnAUSBKey; остальная часть нашего руководства будет относиться и к другим дистрибутивам тоже, и мы предполагаем, что у вас имеется базовая установка с рабочим интернет-соединением. Неплохо было бы настроить на вашем компьютере демон SSH (на случай, если что-то пойдет не так) и статический IP-адрес. Эти шаги хорошо документированы, и мы будем считать, что вы с ними справились. Итак, отключайте монитор и клавиатуру и продолжите создавать устройство удаленно.



Наш массив вполне справился с тестом произвольной записи AIO, а с IOZone вообще замечательно.

Сначала нужно разбить свои диски на разделы. Если эти диски более 2,2 ТБ, следует использовать таблицу разделов GPT. Даже если они не больше, все равно можете сделать именно так. Здесь вам поможет программа *gdisk*, это часть пакета *gptfdisk* в Arch:

```
# gdisk /dev/sda
```

Создайте новый раздел, введя n, затем опять нажмите на Enter, приняв, что это — первый раздел, и нажмите на Enter еще раз, чтобы принять сектор загрузки по умолчанию [2048]. Неплохо бы в конце каждого диска оставить не менее 100 МБ »

Выбор компонентов

Вам не стоит особо переживать по поводу другого оборудования, кроме дисков. Компьютеру незачем быть мощным, ему не нужна затейливая графика, и если вы не собираетесь использовать ZFS (см. стр. 46), 4 ГБ ОЗУ будет более чем достаточно. Популярное решение — HP Microservers, но у них не самые элегантные корпуса. И ведь это так приятно — создавать что-то самим. Возможно, у вас завалялся где-нибудь корпус micro ATX, а если нет, вы вполне сможете собрать весьма симпатичный mini ITX, не входя в особые расходы.

Если ваша машина будет работать 7/24 в вашей гостиной, то вам, вероятно, понадобятся тихие компоненты. И позаботьтесь, чтобы воздух нормально циркулировал вокруг дисков, не давая им перегреваться.

Мы-то, однако, выбрали AMD Kabini 5350 APU (четырехъядерный, 2,05 ГГц, видеокарта R3). Серия Kabini, предназначенная для быстрорастущих малобюджетных рынков, была запущена в апреле, и отличается скромной рассеиваемой мощностью 25 Вт, так что перегрев не должен быть проблемой.

Контроллер на чипе имеет встроенную поддержку только двух дисков SATA, однако карты с 2 портами PCIExpress довольно дешевые. У вас должна быть такая, которая поддерживает переключение на основе FIS (т. е. избегайте всего на основе чипа ASM1061). Если вы предпочитаете крутые чипы, то по качеству и цене вам подойдет Celeron за £190. Есть немало материнских плат Mini-ITX со встроенным процессором. Как и платы AM1, некоторые допускают питание от стандартного 19-В блока для ноутбука.

свободными, поскольку диски, заявленные на одинаковую емкость, часто на пару цилиндров различаются. Здесь вы можете либо прибегнуть к вычислениям, чтобы выяснить, каким именно сектором закончить раздел (умножьте размер вашего диска в терабайтах на 2 в степени 40, 100 раз вычтите 2 в степени 20, поделите на 512 — каждый сектор, вероятно, 512 байт — добавьте 2048, бум...), либо просто использовать, например, [b] +5999.9G [/b] для, допустим, 100 мегов, которых не хватает до 6 ТБ. Разделам RAID надо присваивать специальный тип, FD00, хотя Linux вообще-то перестал обращать на это внимание. Запишите новую таблицу разделов на диск, введя w. Повторите эти действия для всех дисков, которые хотите включить в свой массив.

Настройка массива

Самая интересная и самая длительная по времени часть — настройка массива. Главные сложности вы обойдете на уровне абстракции *mdadm*, однако позаботьтесь о правильности параметров: указанные вами разделы будут безвозвратно очищены от данных.

Например, наш массив RAID6 ожил после следующего заклинания:

```
# mdadm --create --verbose –level=6
--metadata=1.2 --chunk=256 --raid-devices=4 /
dev/md0/ dev/sda1 /dev/sdb1 /dev/sdc1 /dev/sdd1
```

Команда будет долго-долго работать в фоновом режиме (на создание нашего массива ушло 24 часа), и вы можете отслеживать прогресс по файлу состояния:

```
# cat /proc/mdstat
```

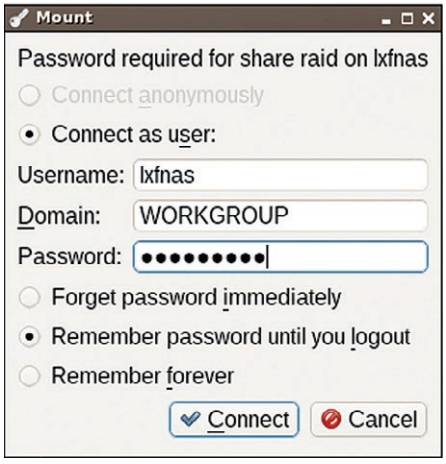
В режиме ограниченной функциональности можно начать работу со своим массивом немедленно, но раз терпение — это добродетель, не лучше ли пока почитать книгу или выпить пару чашек чая? Или погрузитесь в раздумья, а не выбрать ли за объем порции данных [chunk] (то есть тех частей, по каким ваши данные распределяются по дискам) 256K. По умолчанию это 512K, но оптимальный объем зависит от вашего оборудования и его применения. Для более крупных файлов рекомендуются более мелкие порции, чтобы раскидать данные по большему числу дисков, но при малом количестве дисков эта логика не работает. Для более мелких файлов стоит использовать порции покрупнее, но уж не на порядок больше объема файлов, с которыми вы работаете. Если вы намерены отыскать оптимальное соотношение, придется потратить несколько часов на сравнительные тесты вашей системы. Помните: даже если доступ к NAS у вас через Gigabit Ethernet, узким местом, вероятнее всего, будет сеть, так что в этом смысле особо тонко настраивать параметры RAID означает заниматься ерундой. Однако заданный параметр будет важен при инициализации нашей файловой системы.

```
Conveyance self-test routine
recommended polling time:        (   5) minutes.
SCT capabilities:              (0x303d) SCT Status supported.
                                        SCT Error Recovery Control supported.
                                        SCT Feature Control supported.
                                        SCT Data Table supported.

SMART Attributes Data Structure revision number: 16
Vendor Specific SMART Attributes with Thresholds:
ID# ATTRIBUTE_NAME          FLAG     VALUE WORST THRESH TYPE      UPDATED  WHEN_FAILED RAW_VALUE
  1 Raw_Read_Error_Rate     0x002f   200   200   051    Pre-fail  Always       -       0
  3 Spin_Up_Time            0x0027   206   203   021    Pre-fail  Always       -       8691
  4 Start_Stop_Count        0x0032   100   100   000    Old_age   Always       -       15
  5 Reallocated_Sector_Ct   0x0033   200   200   140    Pre-fail  Always       -       0
  7 Seek_Error_Rate         0x002e   200   200   000    Old_age   Always       -       0
  9 Power_On_Hours          0x0032   100   100   000    Old_age   Always       -       48
 10 Spin_Retry_Count        0x0032   100   253   000    Old_age   Always       -       0
 11 Calibration_Retry_Count 0x0032   100   253   000    Old_age   Always       -       0
 12 Power_Cycle_Count       0x0032   100   100   000    Old_age   Always       -       15
192 Power-Off_Retract_Count 0x0032   200   200   000    Old_age   Always       -       5
193 Load_Cycle_Count        0x0032   200   200   000    Old_age   Always       -       19
194 Temperature_Celsius     0x0022   123   110   000    Old_age   Always       -       29
196 Reallocated_Event_Count 0x0032   200   200   000    Old_age   Always       -       0
197 Current_Pending_Sector  0x0032   200   200   000    Old_age   Always       -       0
198 Offline_Uncorrectable   0x0030   100   253   000    Old_age   Offline      -       0
199 UDMA_CRC_Error_Count    0x0032   200   200   000    Old_age   Always       -       0
200 Multi_Zone_Error_Rate   0x0008   100   253   000    Old_age   Offline      -       0

SMART Error Log Version: 1
No Errors Logged
```

› *Smartmontools* могут учесть данные SMART вашего диска и предсказать проблемы, грозящие оборудованию.



› С задачами монтирования SMB управится *Qt*-порт *PcManFM*. Он даже не будет приставать к вам с паролем для гостевых разделов.

Надо сообщить *mdadm* о своем массиве, чтобы обеспечить легкий доступ к нему после загрузки. Сделайте это, запустив

```
# mdadm --detail --scan >> /etc/mdadm.conf
```

что добавит к файлу настройки *mdadm* примерно такую строку:

```
ARRAY /dev/md0 metadata=1.2 name=wdarray:0
UUID=35f2b7a0:91b86477:b ff71c2f:abc04162
```

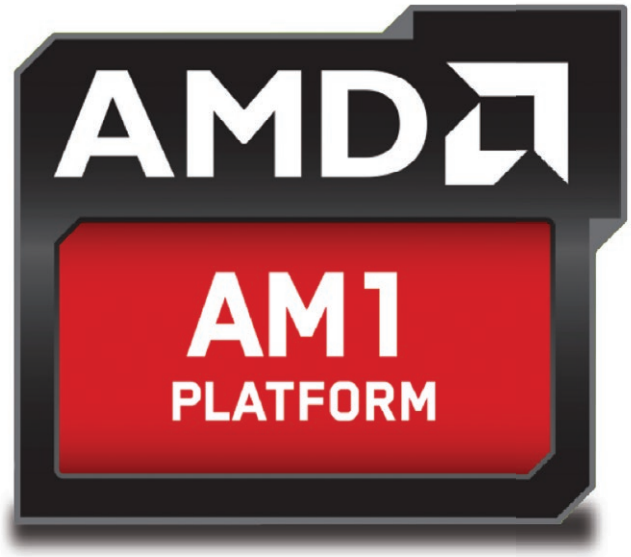
Теперь с узлом устройства **/dev/md0** можно обращаться как с любым другим разделом, пусть даже в нашем случае это увесистые 12 ТБ. Давайте же отформатируем его, готовя к массивному притоку данных. Мы применим ext4, что, вероятно, покажется консервативным выбором, но это надежная и современная файловая система. Более экзотические варианты хорошо масштабируются на десятки дисков и могут даже управлять вашим массивом независимо от *mdadm*, но ZFS требует много памяти (настоятельно рекомендуется дорогая память ECC), а Btrfs может привести к всплескам нагрузки на CPU. Ext4 имеет пару специальных опций RAID — **stride** и **stripe-width**, и важно правильно их понимать, чтобы выровнять

Внимание: RAID

Это предупреждение красуется в любой статье о RAID, но мы все равно добавим его и в нашу: RAID — это не резервное копирование! Это лишь первая линия обороны, и она не защитит вас от случайного удаления информации. (Как и от пожара, электромагнитных импульсов и черной магии.) Если ваши данные действительно важны, сделайте их резервную копию на внешнем носителе. Наше руководство показывает, как настроить программный RAID, и стоит развенчать ряд мифов на эту тему.

Есть специальные контроллеры RAID, хотя и по более высокой цене. По большей части особой нужды в них нет — конечно, расчет битов четности малость нагружает процессор, но на современном железе это пренебрежимо мало. Кроме того, контроллеры оборудования обычно используют проприетарные структуры диска, и если ваш контроллер откажет, придется заменить его идентичным, чтобы добраться до своих дисков. Доступ к программному массиву RAID можно получить с помощью любой ОС Linux через команду **mdadm**. Аппаратные контроллеры RAID бывают придирчивы к совместимости дисков SATA; а при программном RAID, если ОС видит диск, то управится и с RAID.

И, наконец, ваша материнская плата может заявить, что она поддерживает различные конфигурации RAID. Это именуется FakeRAID [ложный RAID], или иногда host RAID. Несмотря на слегка презрительное название (встроенный контроллер перекладывает все вычисления RAID на CPU), это все же стабильная структура (хотя обычно поддерживает только RAID 0, 1 и 10), позволяющая порционно распределить ваш загрузочный диск и иногда даже восстановить ваш массив из BIOS. Однако восстановление может потребовать применения программ Windows. Увы, но это так.



AMD APU Athlon 5350 — дешевый и бойкий, и более чем мощный для скромного устройства NAS.

24 ТБ отличного хранилища Western Digital Red. Емко.

порции и блоки файловой системы RAID. Шаг [Stride] — это количество блоков файловой системы в каждой порции, вычисляемое как размер порции/блок. Ext4 по умолчанию использует блоки 4K (хотя можно указать величину поменьше через опцию **-b**), так что каждая из наших порций объемом 256K содержит 64 блока. Ширина шага — это данная цифра, помноженная на количество дисков с данными. В нашем массиве из четырех дисков каждая порция распределена по двум дискам, а два других отведены под четность порции; таким образом, наша ширина шага составляет 128 блоков. Используй мы RAID5, диском четности был бы только один диск, но мы его не используем, поэтому отформатировали массив следующим образом:

```
# mkfs.ext4 -v -L wdarray -m 0.5 -E
stride=64,stripe-width=128 /dev/md0
```

Опция -m настраивает долю раздела в процентах, отведенную для суперпользователя. По умолчанию это 5%, что для более крупных томов многовато.

Samba: потанцуем?

Нам нужно добавить как минимум одного пользователя — дадим ему имя lxfraid — и изменить разрешения на **/mnt/raid**, выдав ему доступ к нашим данным:

```
# groupadd raidusers
# useradd -m -G raidusers -s /bin/bash lxfraid
# chown root:raidusers /mnt/raid
# chmod 775 /mnt/raid
```

Теперь можно устанавливать все необходимые пакеты:

```
# pacman -S samba
```

Помимо наличия учетной записи в системе, пользователи *Samba* должны иметь запись в файле **smbpasswd**. Это делается с помощью

```
# smbpasswd -a lxfraid
```

Затем почитаем в редакторе файл **/etc/samba/smb.conf**: здесь есть пара вещей, которые вы, возможно, решите включить или выключить. Сначала раскомментируйте и отредактируйте строку hosts allow, чтобы ограничить доступ к *Samba* локальной сетью (например, 192.168. или 192.168.0.) и внутренним интерфейсом. Ближе к концу файла вы найдете раздел Share Definitions. Добавьте следующий блок, с данными для нашего массива:

```
[raid]
   comment = LXF RAID
   path = /mnt/raid
   public = no
   valid users = lxfraid
   writable = yes
```

Позднее вы сможете добавить и других, включить их в группу raidusers и настроить для них разрешения. Желая настроить публичную область, где не требуется аутентификация, сначала создайте директорию:

```
# mkdir /mnt/raid/public
```

По умолчанию непривилегированному пользователю nobody приписывается гостевой доступ, а если требуется доступ на запись [write], надо сделать в этой директории **chmod 777** и раскомментировать последнюю строку в следующем определении:

```
[public]
   comment = Guest area
   path = /mnt/raid/public
   public = yes
   read only = yes
   ;writable = yes
```

Альтернатива, разрешающая доступ на запись только пользователям из группы raidusers —

```
# chown lxfraid:raidusers
# chmod 775 /mnt/raid/public
```

плюс добавление в определение [public] строки

```
writelist = @raidusers
```

Теперь мы можем запустить сервисы *Samba* и проверить наш сервер:

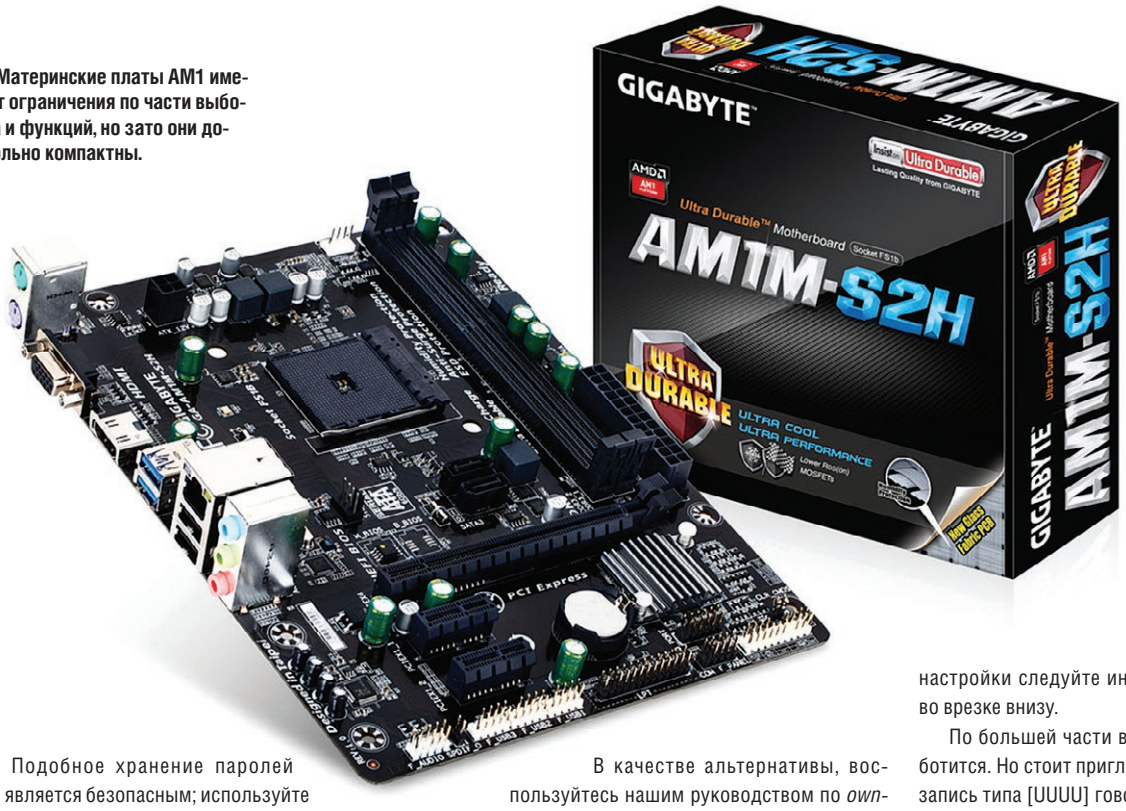
```
# systemctl start {smbd,nmbd}
```

У вас должен быть доступ к разделам *Samba* из любого места вашей сети. Сервис *nmbd* позволит найти ваши разделы *Samba* через URI \\hostname (Windows) или smb://hostname (Mac/Linux). Однако при этом возможны капризы, и удачнее будет обойтись IP-адресами. *Nautilus* и *Dolphin* позволят вам просматривать такие ресурсы из раздела Network. Машинам, которым нужно видеть разделы SMB, следует как минимум установить пакет *smbclient*, чтобы просматривать и монтировать сетевые разделы. Если вам не удалось получить доступ к своим разделам, тогда удачи вам в поиске проблем — неплохим началом будет команда **testparm**, она проверит ваш **smb.conf** на предмет аномалий. Затем можете настроить эти сервисы на автоматический запуск, заменив **start** на **enable**.

> «Главные сложности вы обойдете на уровне абстракции mdadm.»

Если вы хотите автоматически монтировать свой сервер с другой машины в сети, можно добавить в **/etc/fstab** нечто вроде

```
//192.168.133.225/raid /mnt/raid cifs
username=lxfraid, password=password 0 0
```



› Материнские платы AM1 имеют ограничения по части выбора и функций, но зато они довольно компактны.

Подобное хранение паролей не является безопасным; используйте его только на тех компьютерах, где подсматривать некому. Кроме того, если, в силу любых причин сервис *Samba* не работает — скажем, компьютер выключен — то на машине с подобной записью в **fstab** запуск будет замедленным.

Открываемся миру

Иногда желательно сделать свой NAS доступным через Интернет. К этому не стоит относиться легко, и это отнюдь не стоит делать с помощью *Samba*. Одна из возможностей — перенаправить порт 22 на вашем роутере на ваш NAS и соединяться через SFTP. Вы должны защитить SSH-сервер — в частности, неплохо использовать строку

```
PermitRootLogin without-password
```

Это ограничит доступ root только публичным ключом, так что вам придется сгенерировать этот ключ через **ssh-keygen** и добавить его в файл **/root/.authorized_keys**.

Можно также отключить и логины root, установив вышеупомянутую опцию в **no**, однако тогда придется выполнять все задачи администрирования локально, то есть подключить к устройству NAS клавиатуру и монитор.

В качестве альтернативы, воспользуйтесь нашим руководством по *ownCloud* [Учебники **LXF190**, стр. 64] и добавьте раздел *Samba* во внешнем хранилище.

Поскольку ваш внешний IP-адрес, вероятно, изменится, вам нужно настроить динамический DNS с помощью такого сервиса, как DuckDNS, dyndns или no-ip. Эти сервисы позволяют запускать на вашей машине скрипт или клиентскую программу, которая будет обновлять схему DNS для вашего IP-адреса. Настроив процедуру *cron*, чтобы она периодически запускала ее на вашей машине, вы обеспечите постоянный доступ к своей машине через постоянное доменное имя. Подписаться несложно на любой из этих сервисов, но DuckDNS отличается тем, что позволяет вам всё обновлять посредством простого скрипта, в соответствии с философией Arch — KISS [Keep It Simple, Stupid — «будь проще, дурень»]. Для его настройки следуйте инструкциям, приведенным во врезке внизу.

> «Иногда желательно сделать свой NAS доступным через Интернет.»

По большей части ваш RAID сам о себе позаботится. Но стоит приглядывать за **/proc/mdstat** — запись типа [UUUU] говорит о том, что все работает нормально, а если какой-то диск засбоит, в этом ряду появится буква F. Кроме того, можно добыть информацию из

```
# mdadm --detail /dev/md0
```

Любителям более практического подхода понравится регулярное «патрулирование памяти» своего массива. (В конце концов, в этом нет ничего плохого.) Это проверка на наличие несоответствий между данными и блоками четности, а затем автоматический их ремонт. Запуск данной процедуры —

```
# echo check > /sys/block/md0/md/sync_action
```

Это действие потребует времени — файл **mdstat** будет отслеживать состояние, но если вы в любой момент решите его отменить, скомандуйте

```
# echo idle > /sys/block/md0/md/sync_action
```

С устройством NAS можно сделать еще много интересного, но на данном этапе мы исчерпали отведенный нам лимит. Почему бы вам не написать нам и не рассказать о своих приключениях, связанных с NAS? LXF

Настройка DuckDNS

На сервис DuckDNS можно подписаться на www.duckdns.org через Twitter, Facebook, Reddit или Google+. А если вы — явление нестандартное, придется вам решить, какое из этих четырех зол для вас наименьшее, и создать специальную учетную запись. Перейдите на пользователя lxfraid и создайте наш скрипт обновления DuckDNS:

```
# su lxfraid
$ mkdir ~/duckdns
$ cd duckdns
$ nano duck.sh
```

Введите следующее, соответственно заменив домены и токены.

```
echo url="https://www.duckdns.org/update?domains=your_domain&token=your_token&ip=" | curl -k -o ~/duckdns/duck.log -K -
```

Затем сделайте его исполняемым и запустите, чтобы протестировать:

```
$ chmod 700 duck.sh
$ ./duck.sh
```

Вы должны получить некий результат от *curl*, и, надо надеяться, файл **duck.log** теперь содержит многообещающий текст **OK**. Чтобы запускать сервис автоматически, нужно установить демон *cron* — давайте используем простой *cronie*:

```
# pacman -S cronie
```

Следующая команда (от имени пользователя lxfraid) откроет пустую crontab (в редакторе *nano*, чтобы не вызвать неизбежную ревность *vi*):

```
$ EDITOR=nano crontab -e
```

Добавьте следующую строку, чтобы запускать **duck.sh** каждые пять минут:

```
*/5 * * * * ~/duckdns/duck.sh >/dev/null 2>&1
```