I have pre-emptively replaced you with a very small shell script

Automating all the things.
Pete Stevens
Mythic Beasts Ltd

Job Offer

- I am on the lookout for a Junior Network Bod to help with a data cleanse. Essentially make sure our Config Management DB matches what's on the live network and correct any discrepancies.
- I would imagine it would be a 3 month contract. Anyone interested please get in touch.

Network Audit

- Management hat
  - Employ someone with valuable skills (networking)
  - To do something really boring (auditing)
  - Disruptive (network changes or config management changes to fix)
  - With archeaology (why is prod different to config management?)

Network Audit

- Misses important points
  - How to stop prod drifting from config mangement
  - Without crippling the organisation waiting on a dev team to upgrade the config management system rules
  - 3 months, that’s time for more drift to occur
  - Really want to audit once per week
    - Or once per day
    - Or once per hour

Automating Network Auditing

- Standard tools don't work, e.g. diff
  - 2a00:1098::1 *vs*

    2a00:1098::0:1
  - confederation peers a b c *vs*

    confederation peers b c a
- show running-config might not give you exactly the same format back that you put in
- Some things change all the time (prefix lists)

Automating Network Audit

- We have a series of configuration files that describe (almost) our entire network configuration
- routers.yaml
  - Knows about every network interface and IP address on every router, and the type of interfaces
    - Customer / inter-site / Internet Exchange / Transit
    - Magically works out all the BGP sessions needed
- customers.yaml
  - Knows about every customer that takes BGP from us
    - Work out what customer IP is in which router IP subnet

      and you can magically generate all the bgp sessions

Automating Network Audit

- static.yaml
  - Contains static routes for customers we transit
  - Use this to generate more of our network config
- peers
  - Tab separated file (hystorical reasons)
  - Needs to migrate to a database and get auto-populated from peeringDB
  - Auto build peering config and filters from file + RADB
  - Update peer filters regularly and automatically

Automating Network Audit

- We can now generate all our BGP config for any given router.
- Context aware diff:
  - Canonicalise all IP addresses so text match works
  - Re-order confederations and similar so match works
  - Check prefix lists exist but not contents – prefix lists change too fast

Automating Network Audit

- We can now audit a router config quickly
  - Including building all of our customer prefix lists from RADB, we can audit a router in around 30s
  - Process is now:
    - Audit router
    - Fix config management or production config
    - Re-audit
    - Repeat until no errors.
  - Once no errors, schedule daily audits and fix new errors as they arrive

Benefits of an audited network

- We caught fat finger errors (especially IPv6)
- This one slipped through manual checks for years:
  - Should be:
    - neighbor 2a00:1098:2::5d5d:8563 update-source 2a00:1098:2::5d5d:8566
  - Was actually:
    - neighbor 2a00:1098:2::5d5d:8563 update-source 2a00:1098:2::5d5d:8565
  - The typo’d address meant we didn’t have full v6 redundancy in one site for several years, all v6 went through one router

Benefits of automatically auditing

- We know what config we’re running
- If someone monkey patches we find out quickly
- Doesn’t prohibit an under-duress hotfix
  - Doesn’t let you forget about it!

Not Just Networks

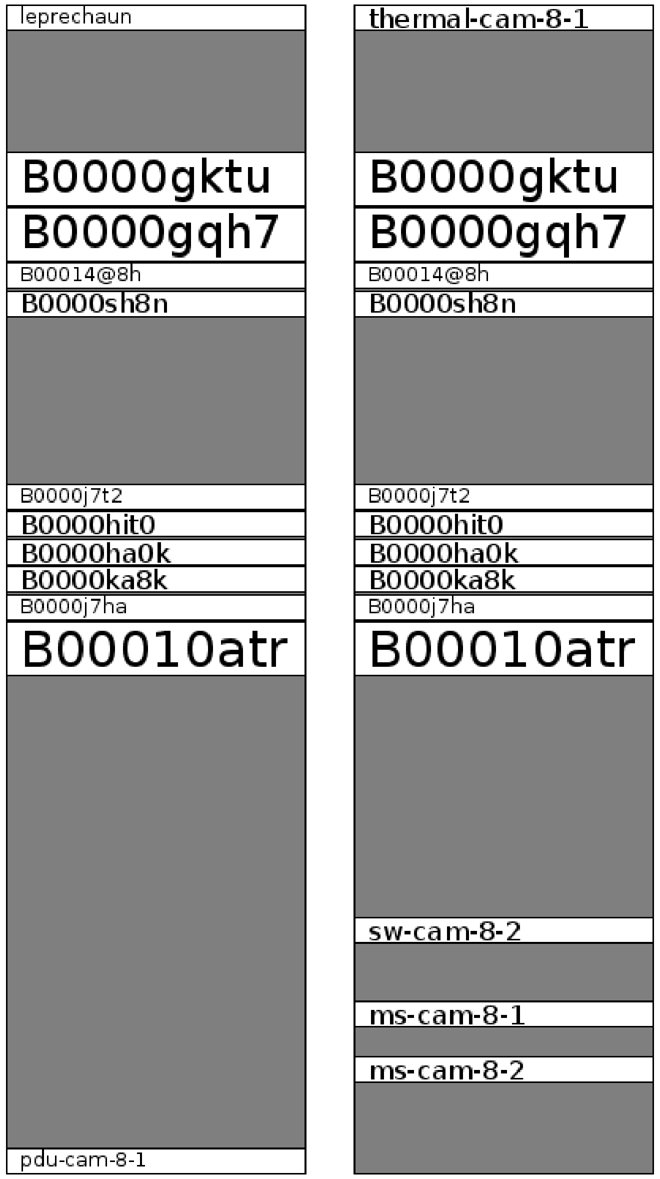
- Billing User <billing@mythic-beasts.com>
- Date: Thu, 04 Apr 2019 10:11:11 +0100
- Unbilled VDSs on vds-ams-a: maryland

- We audit our entire VM estate every day.

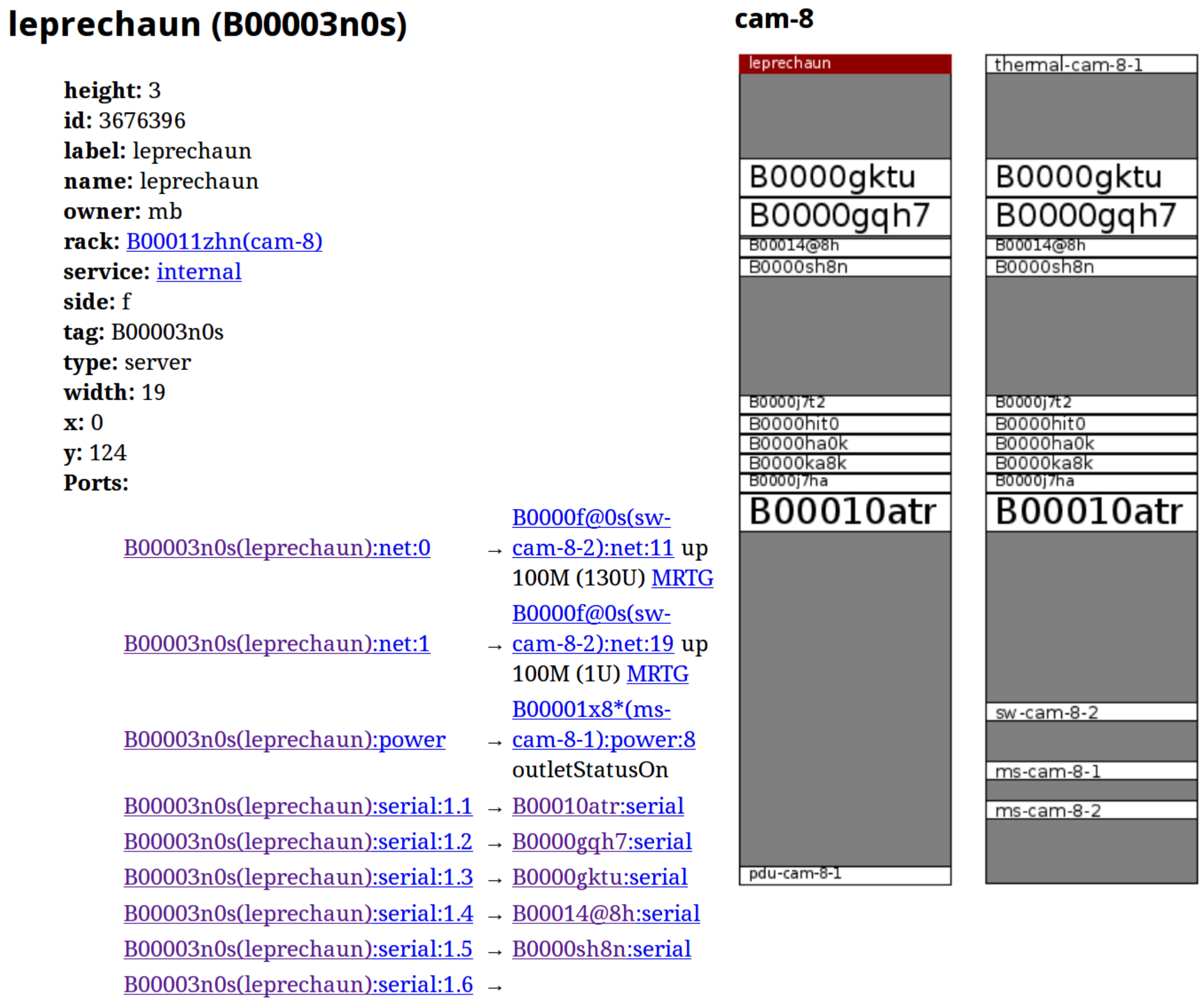
Not Just Networks

- 11:20:08 <richard> anyone know anything about vds:maryland which is on vds-ams-a?
- 11:28:44 <avf> richard, that's not the VDS of the person who wanted his account cancelled, is it?
- 11:29:20 <richard> a/c XXXXX
- 11:29:33 <richard> service cancelled, but evidently not properly
- 11:30:10 <avf> that'd be my fault, I assumed it would be cleaned up automatically

Infrastructure as code



Leprechaun



leprechaun (B00003n0s)

height: 3
id: 3676396
label: leprechaun
name: leprechaun
owner: mb
rack: B00011zhn(cam-8)
service: internal
side: f
tag: B00003n0s
type: server
width: 19
x: 0
y: 124
Ports:

B00003n0s(leprechaun):net:0	→	B0000f@0s(sw-cam-8-2):net:11 up 100M (130U) MRTG
B00003n0s(leprechaun):net:1	→	B0000f@0s(sw-cam-8-2):net:19 up 100M (1U) MRTG
B00003n0s(leprechaun):power	→	B00001x8*(ms-cam-8-1):power:8 outletStatusOn
B00003n0s(leprechaun):serial:1.1	→	B00010atr:serial
B00003n0s(leprechaun):serial:1.2	→	B0000gqh7:serial
B00003n0s(leprechaun):serial:1.3	→	B0000gktu:serial
B00003n0s(leprechaun):serial:1.4	→	B00014@8h:serial
B00003n0s(leprechaun):serial:1.5	→	B0000sh8n:serial
B00003n0s(leprechaun):serial:1.6	→	

Hipsters beware

Our config file format is Perl

```
$tag = 'B00003n0s';

add_asset({

        tag => $tag, height => 3, owner => 'mb', y => $y,

        type => 'server', side => 'f', name => 'leprechaun',

    });

foreach my $port (1 .. 8) {

    add_port ({device => 'leprechaun', type => 'serial', port_index =>
"1.$port" });

    add_port ({device => 'leprechaun', type => 'serial', port_index =>
"2.$port" });

}
```

We call functions in config

```
add_connexion($tag.':power', 'ms-cam-8-1:power:8');
add_connexion($tag.':net:0', 'sw-cam-8-2:net:11');
add_connexion($tag.':net:1', 'sw-cam-8-2:net:19');
```

Auditing

- We can automatically check things from here
- Compile time checks:
  - Do we think a port has more than one device connected?
  - Is each end of a connection connected to something we know about
  - Asset tags are self checksumming, spot typos
  - Does everything that needs a connection have one (e.g. do we know where every power lead is)

More auditing

- Dynamic checks
  - Does every device have a DNS record?
  - Is it powered on?
  - Is the switch port up?
  - Do we have any up switch ports where we don't know what's connected?
- Run a full audit of every site weekly
  - Generates tickets for people to inspect and fix when next on site
  - If someone unplugged everything and put it in a pile, we could put it back
-

Audit will tell you where to plug it in

Cross-system integration

- Automatically build MRTG configuration
  - Power graphs for every power bar
  - Summaries for every rack
  - Graphs for every switch port
  - Summaries for groups of ports (Ixes,Transit)
  - Summaries for racks (power)
  - Entire config built automatically by cron
  - Add a router interface to a transit provider, automagically appears in your graphs

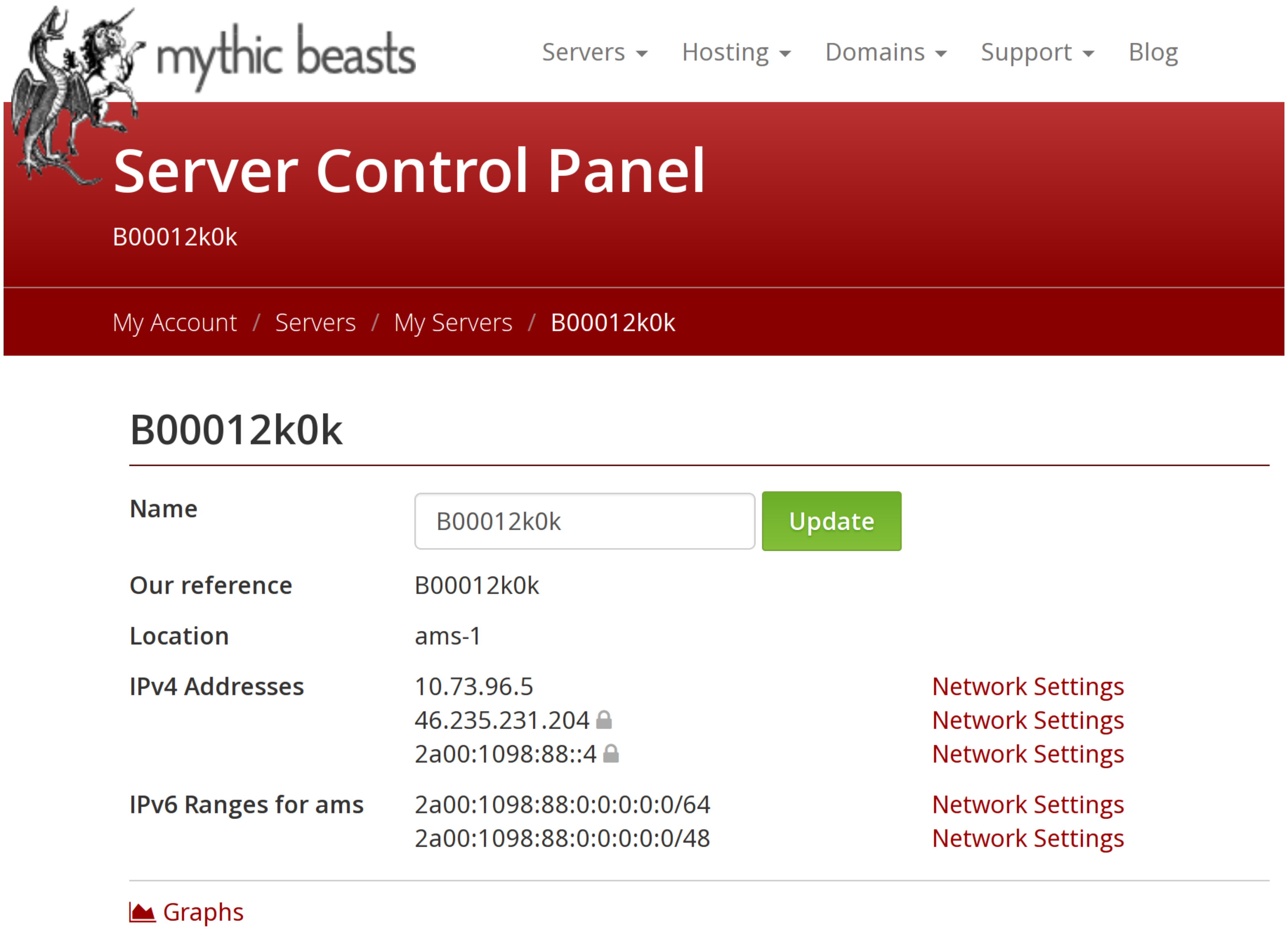
Feeds the control panel

- From an asset tag we can discover automatically
  - Which switch port
  - Which power port
  - Which serial port
  - Network addresses
  - Bandwidth usage
  - Physical location lit up in red on the picture

Billing is key

- Billing attaches things to customers
  - No billing record, no automatic config, service doesn’t work.
  - We learned this in 2007 when we ceased being a part time operation
  - About 1/5th of our customers had never been invoiced.
    - We’d set them up, added a note to invoice later and never done it
    - Same was true for Bluelinux (acquisition in 2012)
    - Bhost also had unbilled VMs they’d forgotten about
    - Due diligence on acquisitions always now involves a physical audit

Control panel populates itself



Server Control Panel

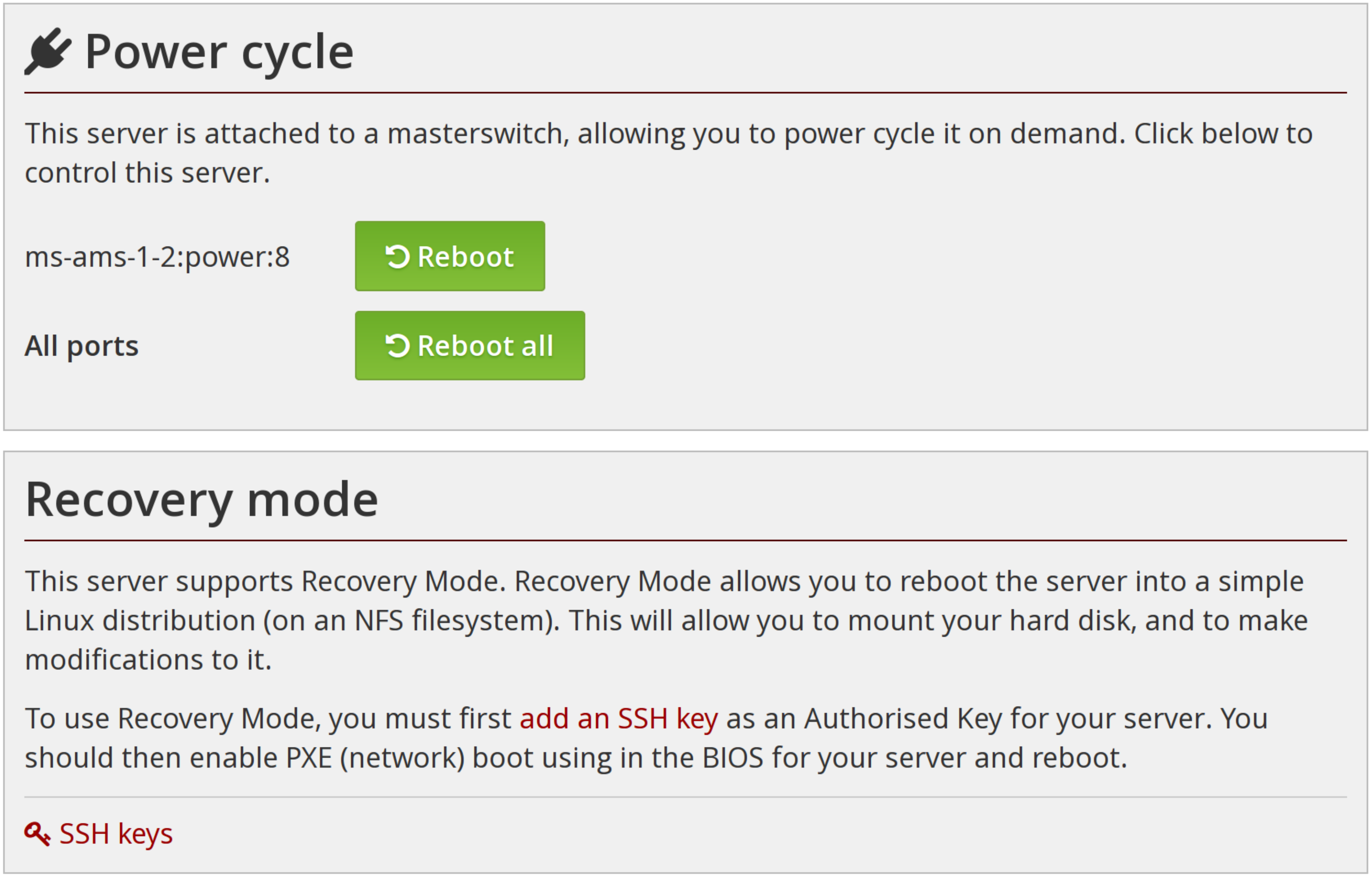
B00012k0k

My Account / Servers / My Servers / B00012k0k

B00012k0k

Name	B00012k0k [Update]	
Our reference	B00012k0k	
Location	ams-1	
IPv4 Addresses	10.73.96.5	Network Settings
	46.235.231.204 🔒	Network Settings
	2a00:1098:88::4 🔒	Network Settings
IPv6 Ranges for ams	2a00:1098:88:0:0:0:0:0/64	Network Settings
	2a00:1098:88:0:0:0:0:0/48	Network Settings

Graphs



Power cycle

This server is attached to a masterswitch, allowing you to power cycle it on demand. Click below to control this server.

ms-ams-1-2:power:8 — Reboot

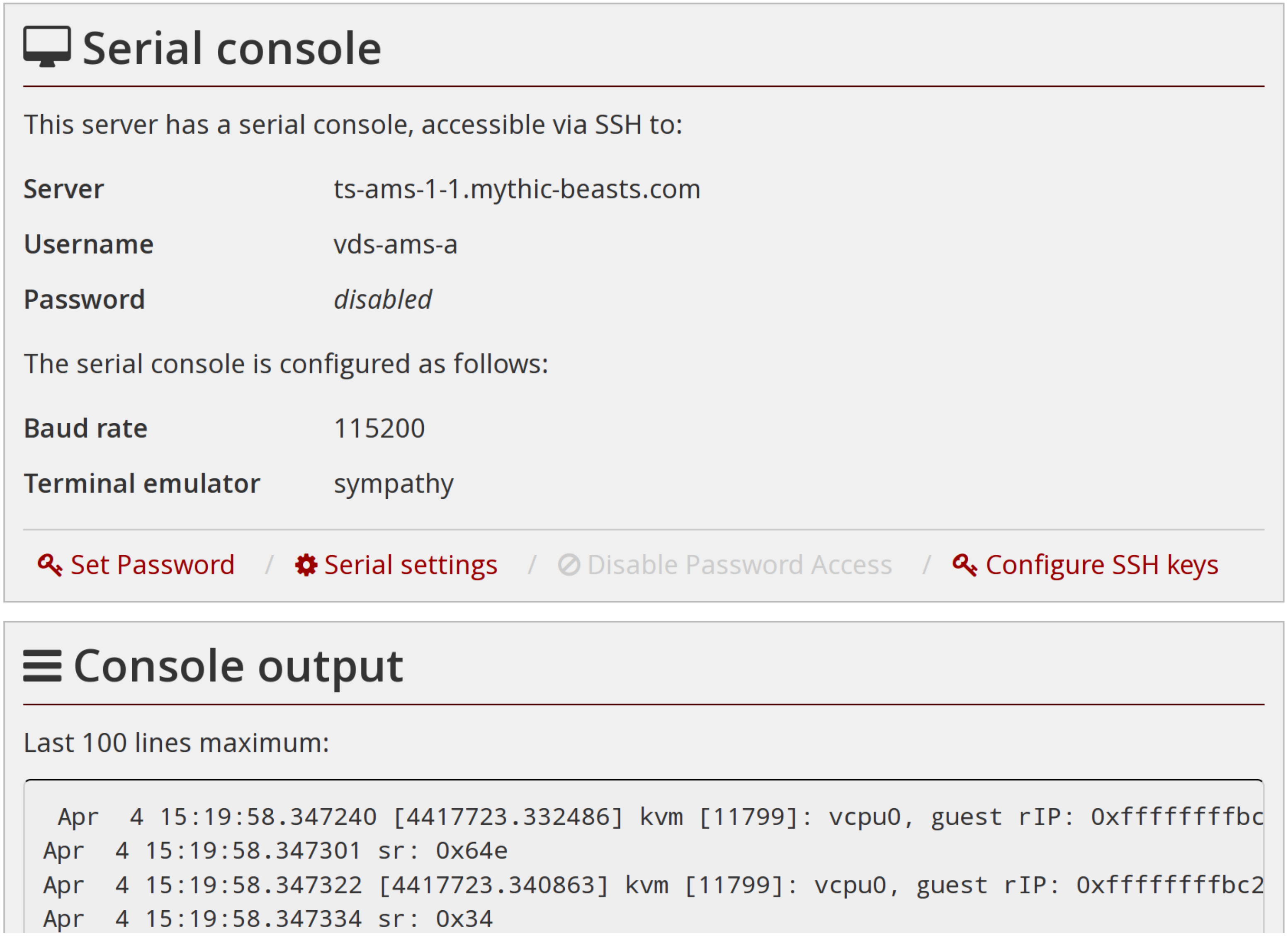
All ports — Reboot all

Recovery mode

This server supports Recovery Mode. Recovery Mode allows you to reboot the server into a simple Linux distribution (on an NFS filesystem). This will allow you to mount your hard disk, and to make modifications to it.

To use Recovery Mode, you must first add an SSH key as an Authorised Key for your server. You should then enable PXE (network) boot using in the BIOS for your server and reboot.

SSH keys



Serial console

This server has a serial console, accessible via SSH to:

Server	ts-ams-1-1.mythic-beasts.com
Username	vds-ams-a
Password	*disabled*

The serial console is configured as follows:

Baud rate	115200
Terminal emulator	sympathy

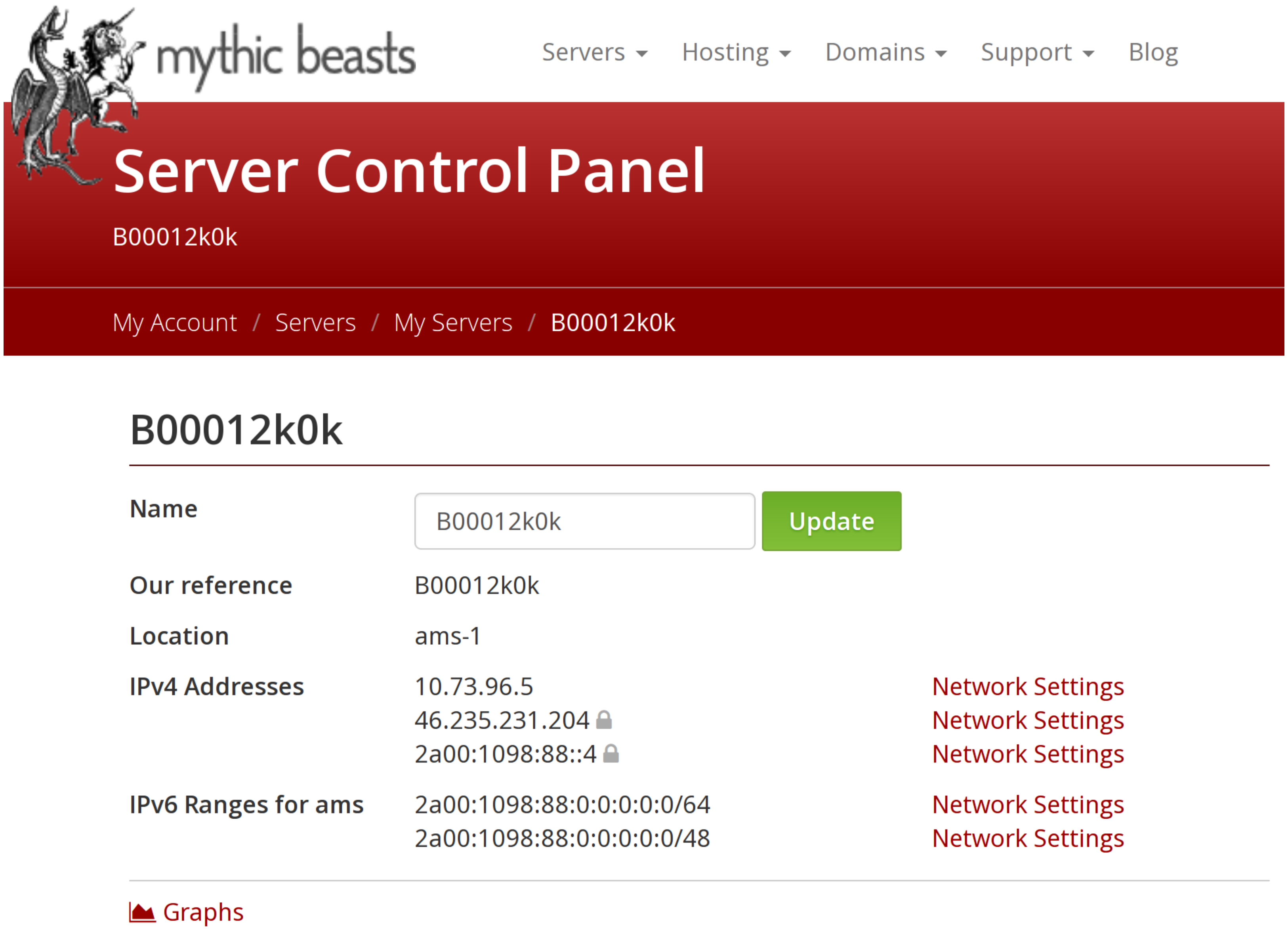
Set Password / Serial settings / Disable Password Access / Configure SSH keys

Console output

Last 100 lines maximum:

```
 Apr  4 15:19:58.347240 [4417723.332486] kvm [11799]: vcpu0, guest rIP: 0xffffffffbc
Apr  4 15:19:58.347301 sr: 0x64e
Apr  4 15:19:58.347322 [4417723.340863] kvm [11799]: vcpu0, guest rIP: 0xffffffffbc2
Apr  4 15:19:58.347334 sr: 0x34
```

Control panel populates itself



Server Control Panel

B00012k0k

My Account / Servers / My Servers / B00012k0k

B00012k0k

Name	B00012k0k [Update]	
Our reference	B00012k0k	
Location	ams-1	
IPv4 Addresses	10.73.96.5	Network Settings
	46.235.231.204 🔒	Network Settings
	2a00:1098:88::4 🔒	Network Settings
IPv6 Ranges for ams	2a00:1098:88:0:0:0:0:0/64	Network Settings
	2a00:1098:88:0:0:0:0:0/48	Network Settings

Graphs

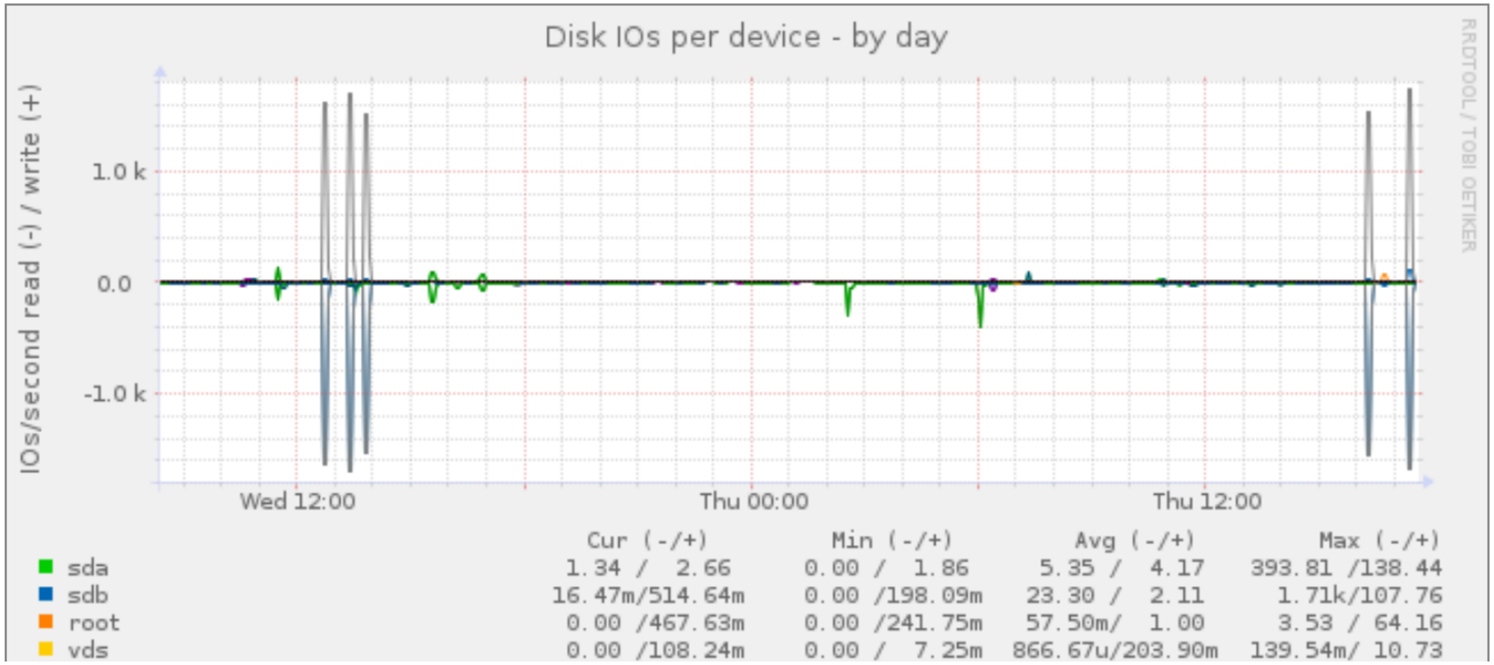
Server graphs

Graphs for B00012k0k

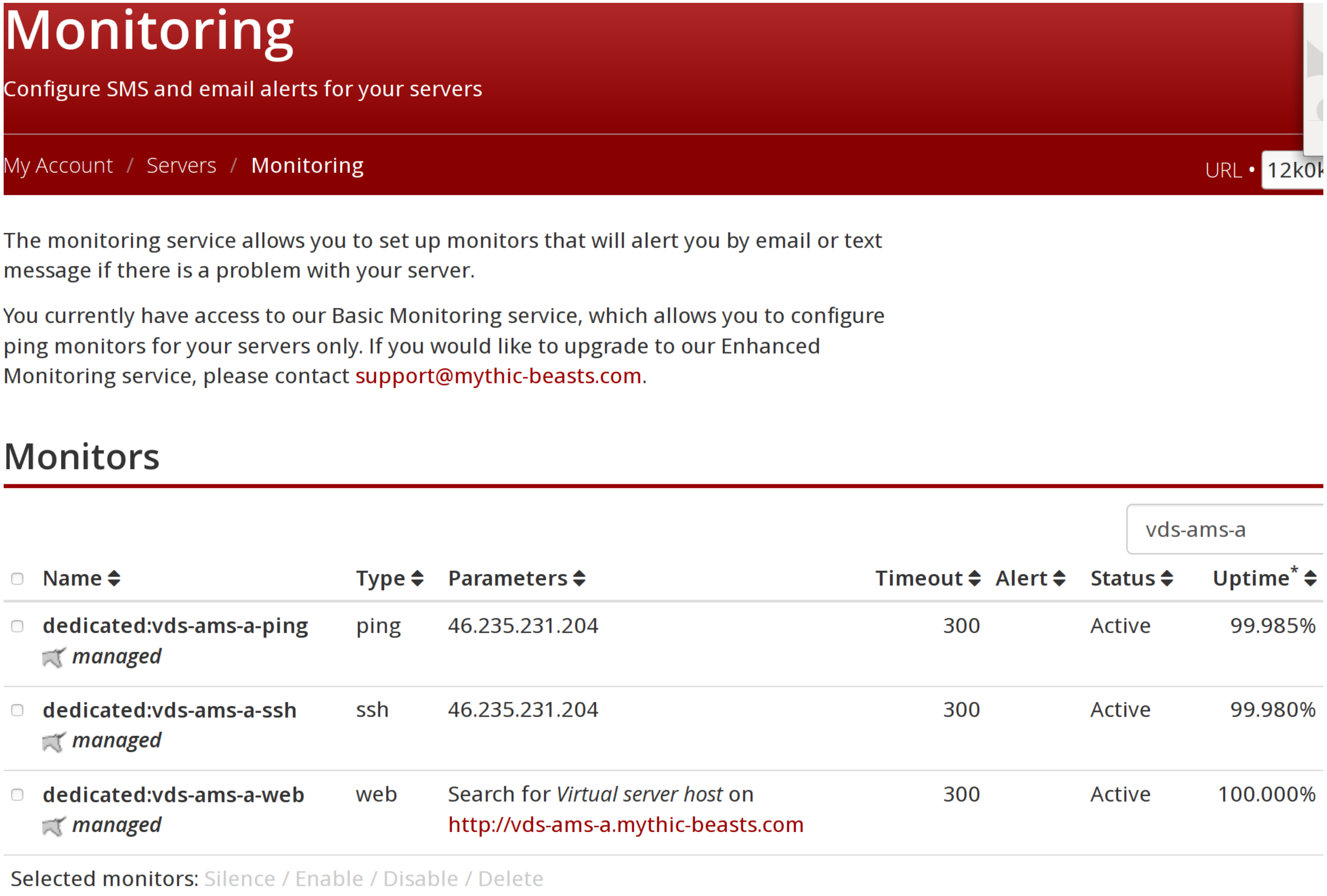
My Account / Servers / My Servers / B00012k0k / **Graphs**

Jump to [disk exim network nfs processes sensors system time]

disk



```
                         Cur (-/+)         Min (-/+)        Avg (-/+)          Max (-/+)
■ sda               1.34 /   2.66     0.00 /   1.86      5.35 /   4.17     393.81 /138.44
■ sdb           16.47m/514.64m     0.00 /198.09m     23.30 /   2.11      1.71k/107.76
■ root              0.00 /467.63m     0.00 /241.75m     57.50m/   1.00      3.53 /  64.16
■ vds               0.00 /108.24m     0.00 /   7.25m   866.67u/203.90m   139.54m/  10.73
```



Monitoring

Configure SMS and email alerts for your servers

My Account / Servers / Monitoring

URL • 12k0k

The monitoring service allows you to set up monitors that will alert you by email or text message if there is a problem with your server.

You currently have access to our Basic Monitoring service, which allows you to configure ping monitors for your servers only. If you would like to upgrade to our Enhanced Monitoring service, please contact support@mythic-beasts.com.

Monitors

vds-ams-a

	Name	Type	Parameters	Timeout	Alert	Status	Uptime*
☐	**dedicated:vds-ams-a-ping** *managed*	ping	46.235.231.204	300		Active	99.985%
☐	**dedicated:vds-ams-a-ssh** *managed*	ssh	46.235.231.204	300		Active	99.980%
☐	**dedicated:vds-ams-a-web** *managed*	web	Search for *Virtual server host* on http://vds-ams-a.mythic-beasts.com	300		Active	100.000%

Selected monitors: Silence / Enable / Disable / Delete

Auto-config

- Munin config – automatic from billing database
- DHCP/recovery config – automatic from database
- Monitoring – port scan all managed servers daily
  - Create ping/ssh/smtp/imap/pop automagically
  - Create ticket for content-aware http/https
  - Ping/ssh created in recovery so uptime not accurate

Internal plumbing

Paul: *“everything should be REST. REST APIs are nice to work with. SSH forced commands always end up as a hideous mess of awk, sed and xargs.”*

Toby: *“everything should be SSH forced commands. We use SSH for admin anyway. REST APIs mean another service to keep running and secure, separate auth config, etc. “*

SSH APIs

- We make extensive use of SSH APIs for internal plumbing:
  - REST API style (hierarchical endpoints, JSON in, JSON out)
  - SSH transport and authentication

SSH API

```
$ ssh -i .ssh/id_rpi_manage pifs-mer-a /unused
{
    "content": {
        "unused": 3
    }
}


$ ssh -i .ssh/id_rpi_manage pifs-mer-a
/rpi/rpi:pete/reboot
{"content":null}
```

Access control

```
root@rpi-fileserver1:~# cat .ssh/authorized_keys

ssh-rsa AAAAB3...WnL Mythic Beasts admin key
mythic-rpihosting

command="/home/rpi/rpi/rpi-manage" ssh-rsa
AAAAB3...3Qf ctrlpanel@billing
```

End point is communicated via $SSH_ORIGINAL_COMMAND

SSH APIs

- Client and server SSH API libraries:
  - Take care of capturing stdout, stderr, exit codes, etc.
- Very quick to develop and deploy
- We use SSH for admin anyway, so all admin access is managed in one place
- SSHFP + DNSSEC avoids the “accept host key” problem when provisioning new servers
  - This is harder than it should be and not yet complete.

Managed Updates

- We have thousands of managed customer servers which we need to security update
- Run a mixture of debian/ubuntu/centos stable versions
- Customers have root and can configure themselves – we can’t use puppet/chef/ansible as we’re not definitive for all configuration
- Cattle not pets is great in theory but we run a cattery

… and a luxury cattery at that!

Managed Updates

- If we see a critical update for apache we want to upgrade just that package now, by running:

```
apt-get install apache2.2
```

or

```
apt-get install apache 2.4
```

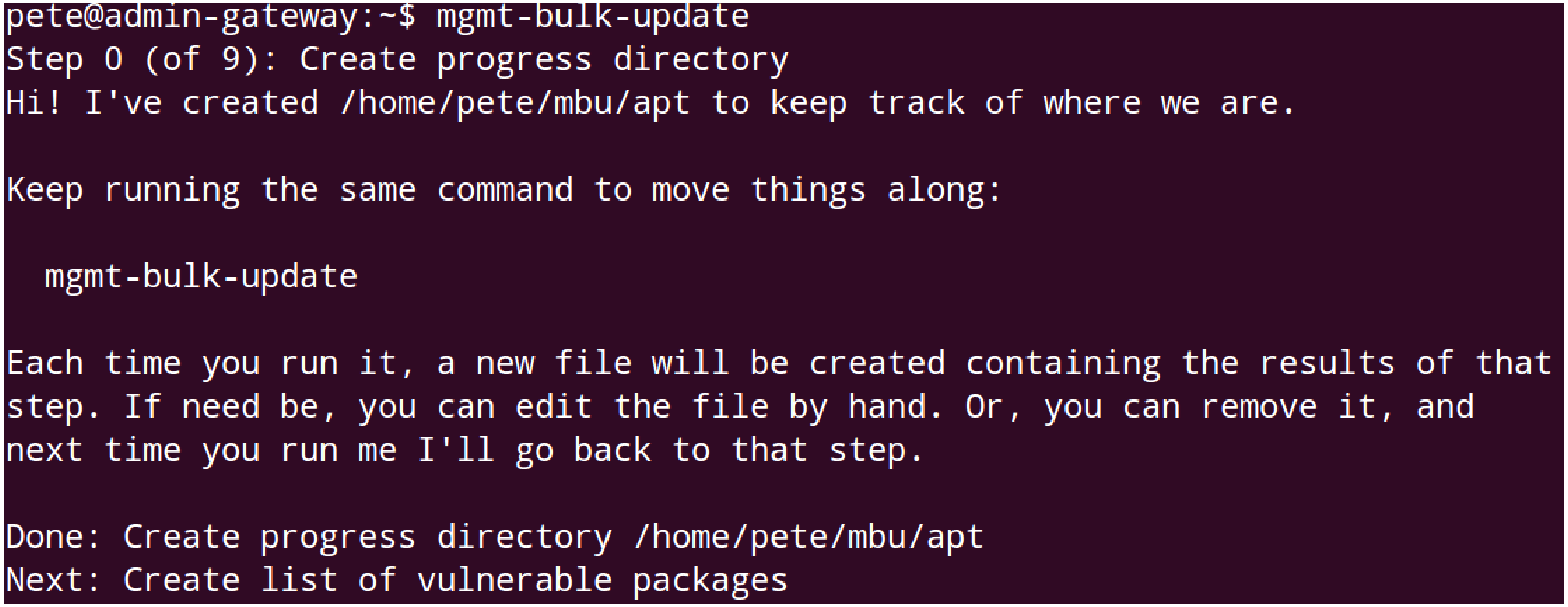
or

```
yum update httpd
```

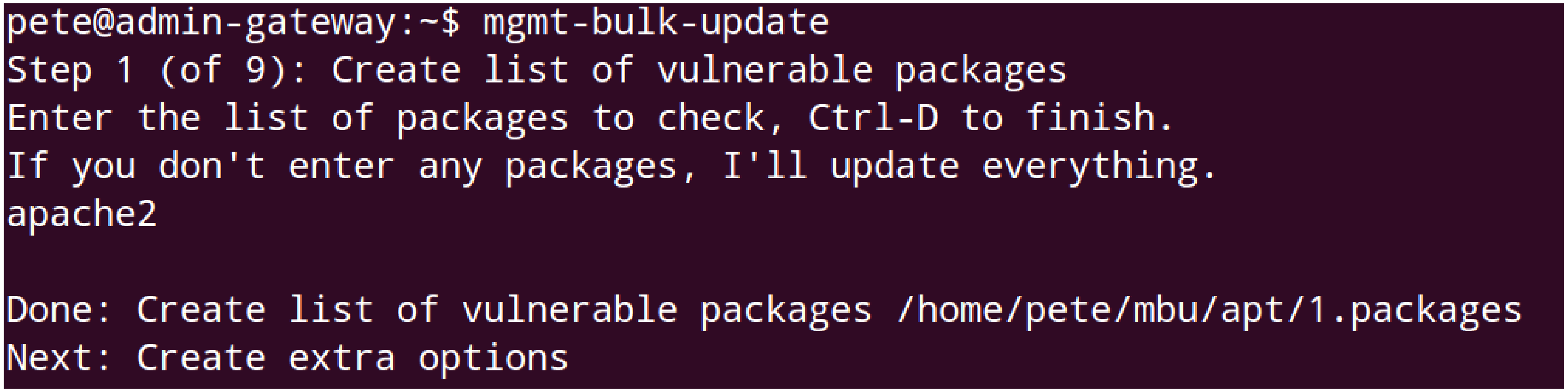
Managed Updates

- We used to install a standard management key on all customer servers, but that’s a security nightmare
- One or more keys per customer
- Private key can’t be accessed by staff, they can only request a session on a customer server
- No agent fowarding, and only the customer-specific key when we log in – minimise risks

Managed bulk update process

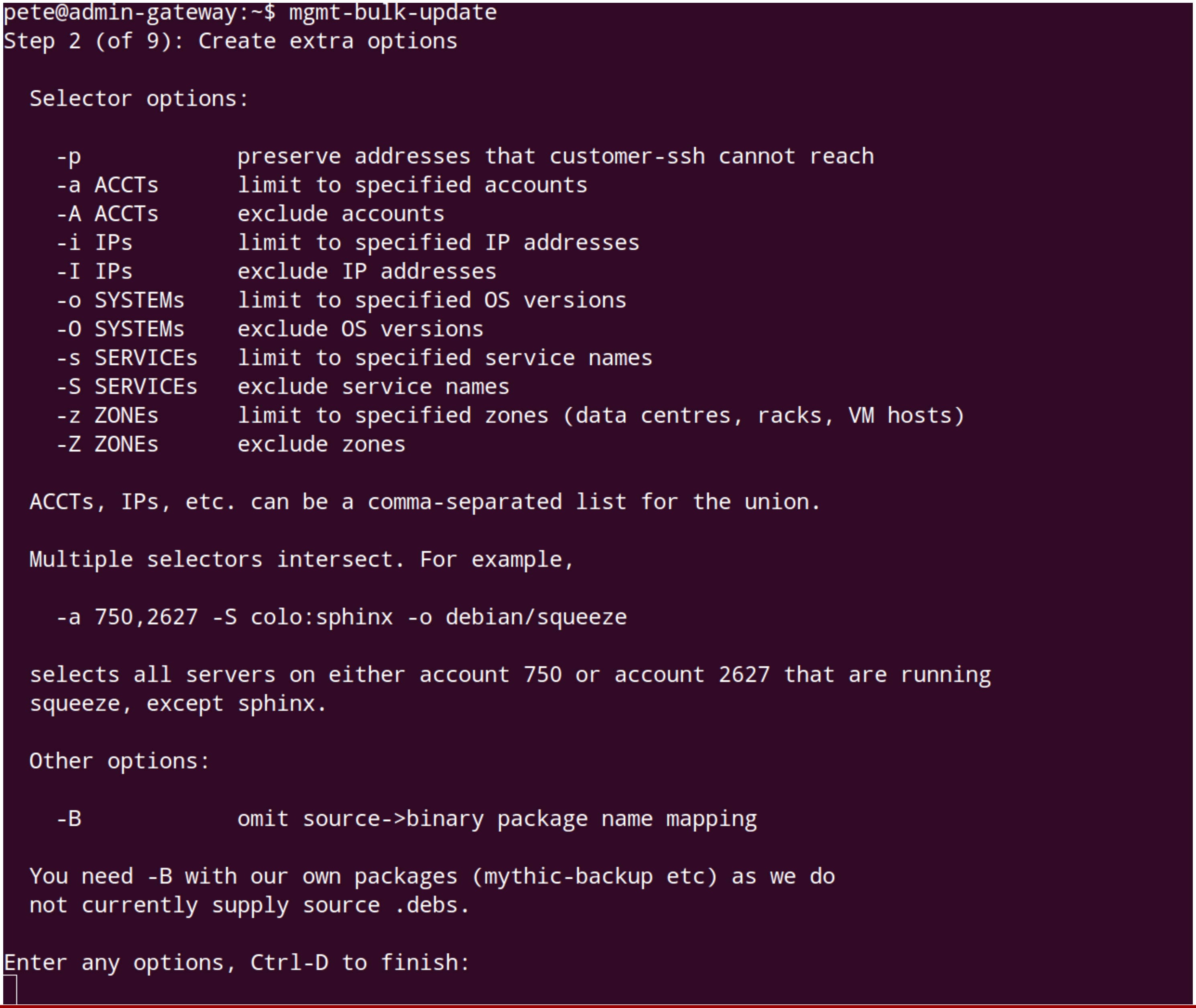


```
pete@admin-gateway:~$ mgmt-bulk-update
Step 0 (of 9): Create progress directory
Hi! I've created /home/pete/mbu/apt to keep track of where we are.

Keep running the same command to move things along:

  mgmt-bulk-update

Each time you run it, a new file will be created containing the results of that
step. If need be, you can edit the file by hand. Or, you can remove it, and
next time you run me I'll go back to that step.

Done: Create progress directory /home/pete/mbu/apt
Next: Create list of vulnerable packages
```



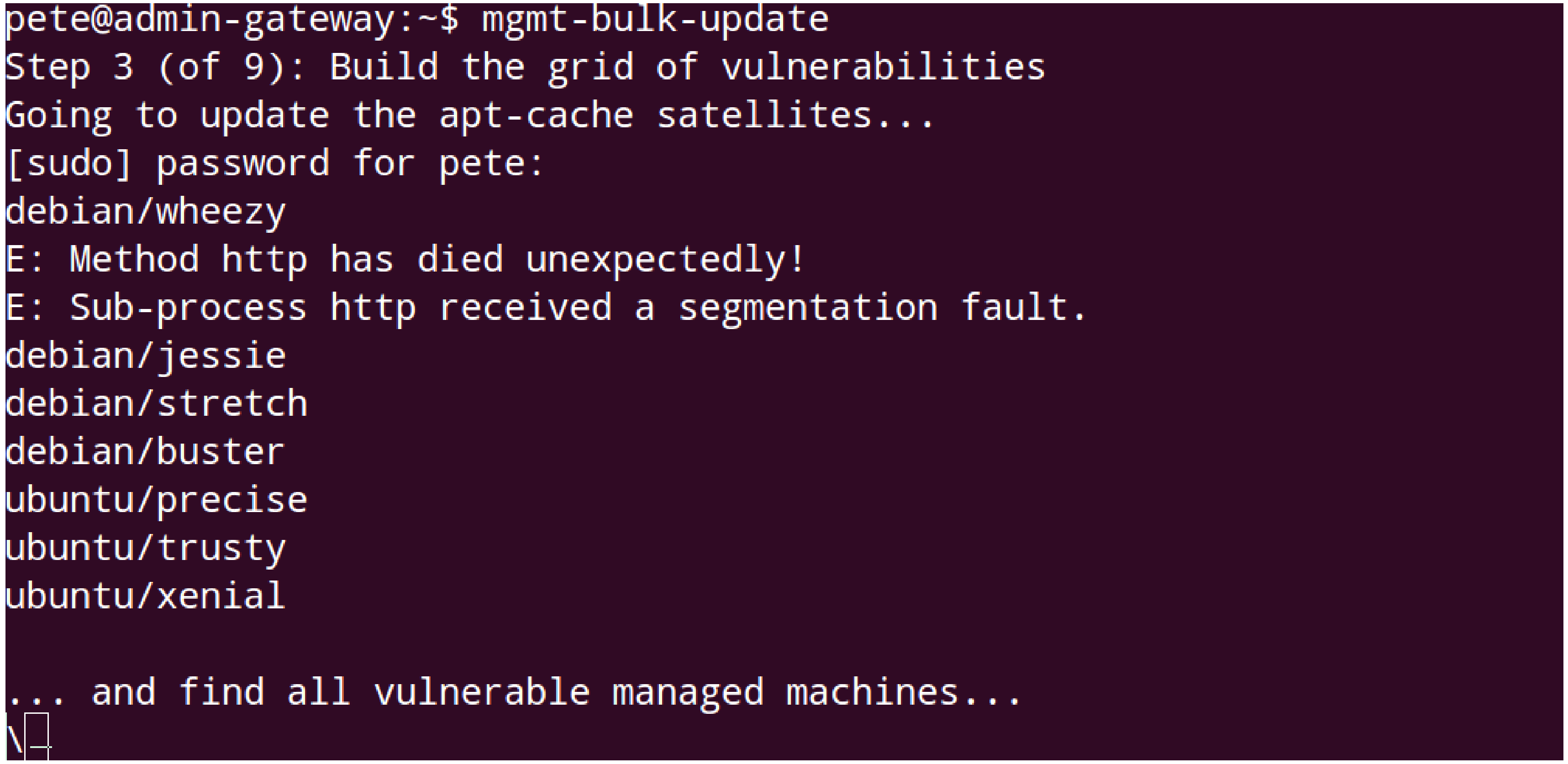
```
pete@admin-gateway:~$ mgmt-bulk-update
Step 1 (of 9): Create list of vulnerable packages
Enter the list of packages to check, Ctrl-D to finish.
If you don't enter any packages, I'll update everything.
apache2

Done: Create list of vulnerable packages /home/pete/mbu/apt/1.packages
Next: Create extra options
```

Managed bulk update process



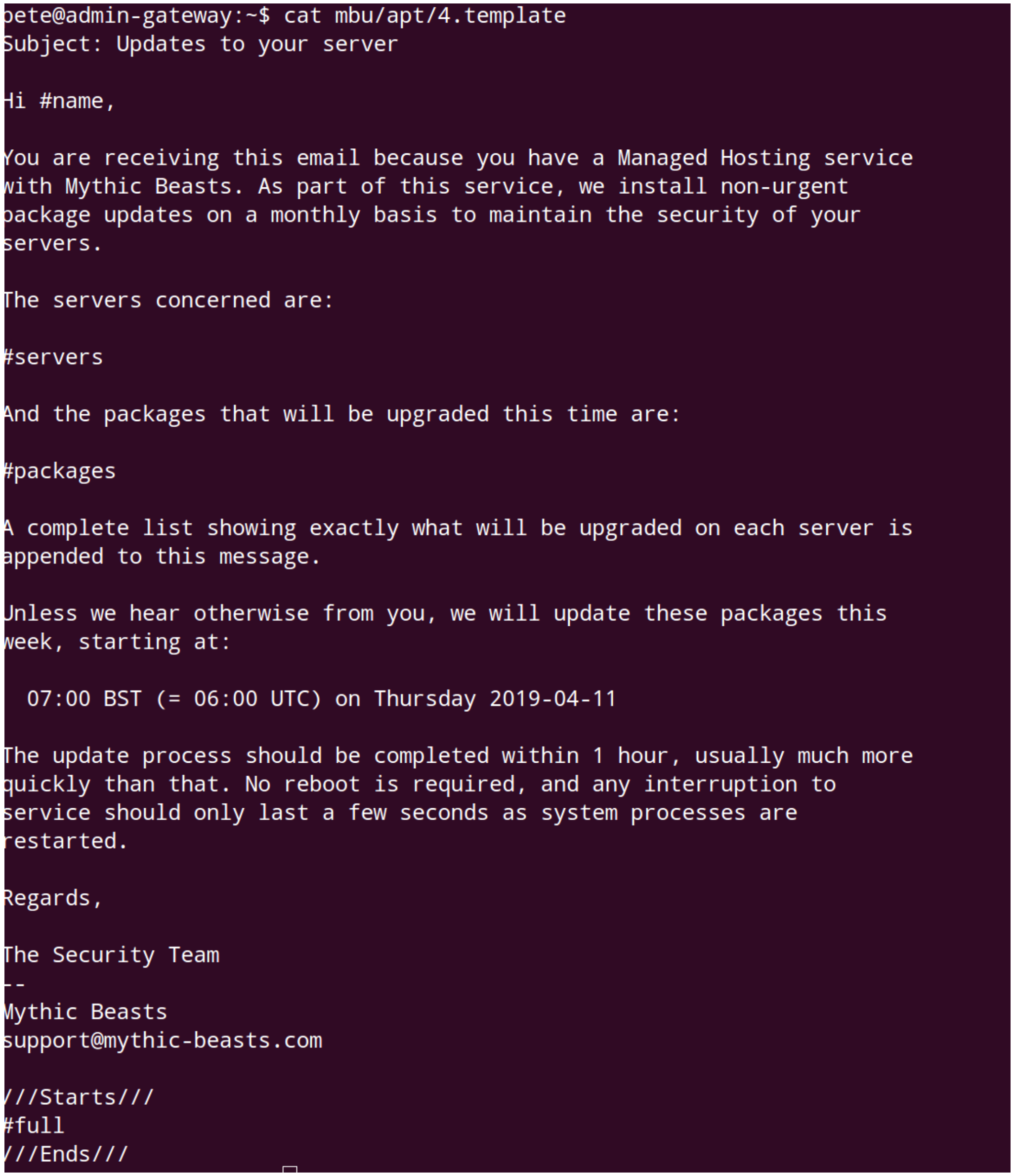
```
pete@admin-gateway:~$ mgmt-bulk-update
Step 2 (of 9): Create extra options

  Selector options:

    -p              preserve addresses that customer-ssh cannot reach
    -a ACCTs        limit to specified accounts
    -A ACCTs        exclude accounts
    -i IPs          limit to specified IP addresses
    -I IPs          exclude IP addresses
    -o SYSTEMs      limit to specified OS versions
    -O SYSTEMs      exclude OS versions
    -s SERVICEs     limit to specified service names
    -S SERVICEs     exclude service names
    -z ZONEs        limit to specified zones (data centres, racks, VM hosts)
    -Z ZONEs        exclude zones

  ACCTs, IPs, etc. can be a comma-separated list for the union.

  Multiple selectors intersect. For example,

    -a 750,2627 -S colo:sphinx -o debian/squeeze

  selects all servers on either account 750 or account 2627 that are running
  squeeze, except sphinx.

  Other options:

    -B              omit source->binary package name mapping

  You need -B with our own packages (mythic-backup etc) as we do
  not currently supply source .debs.

Enter any options, Ctrl-D to finish:
```

Managed bulk update process



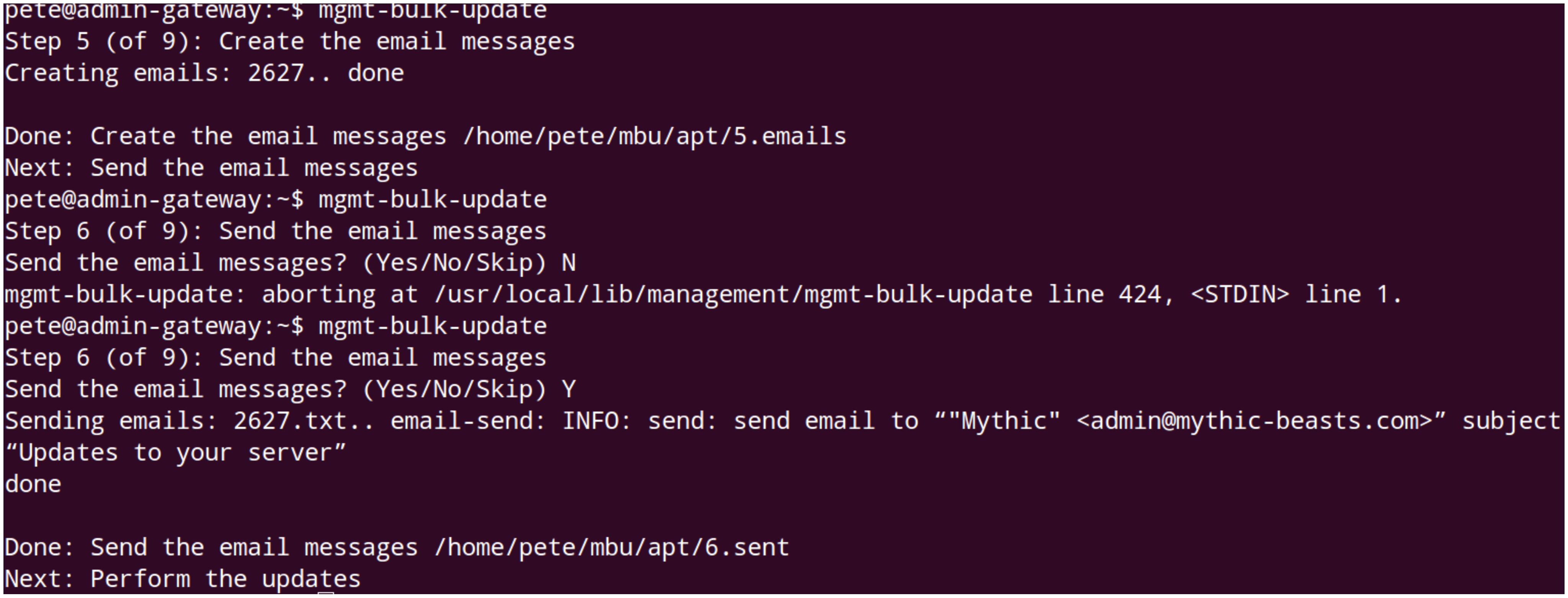
```
pete@admin-gateway:~$ mgmt-bulk-update
Step 3 (of 9): Build the grid of vulnerabilities
Going to update the apt-cache satellites...
[sudo] password for pete:
debian/wheezy
E: Method http has died unexpectedly!
E: Sub-process http received a segmentation fault.
debian/jessie
debian/stretch
debian/buster
ubuntu/precise
ubuntu/trusty
ubuntu/xenial

... and find all vulnerable managed machines...
\
```

Managed bulk update process



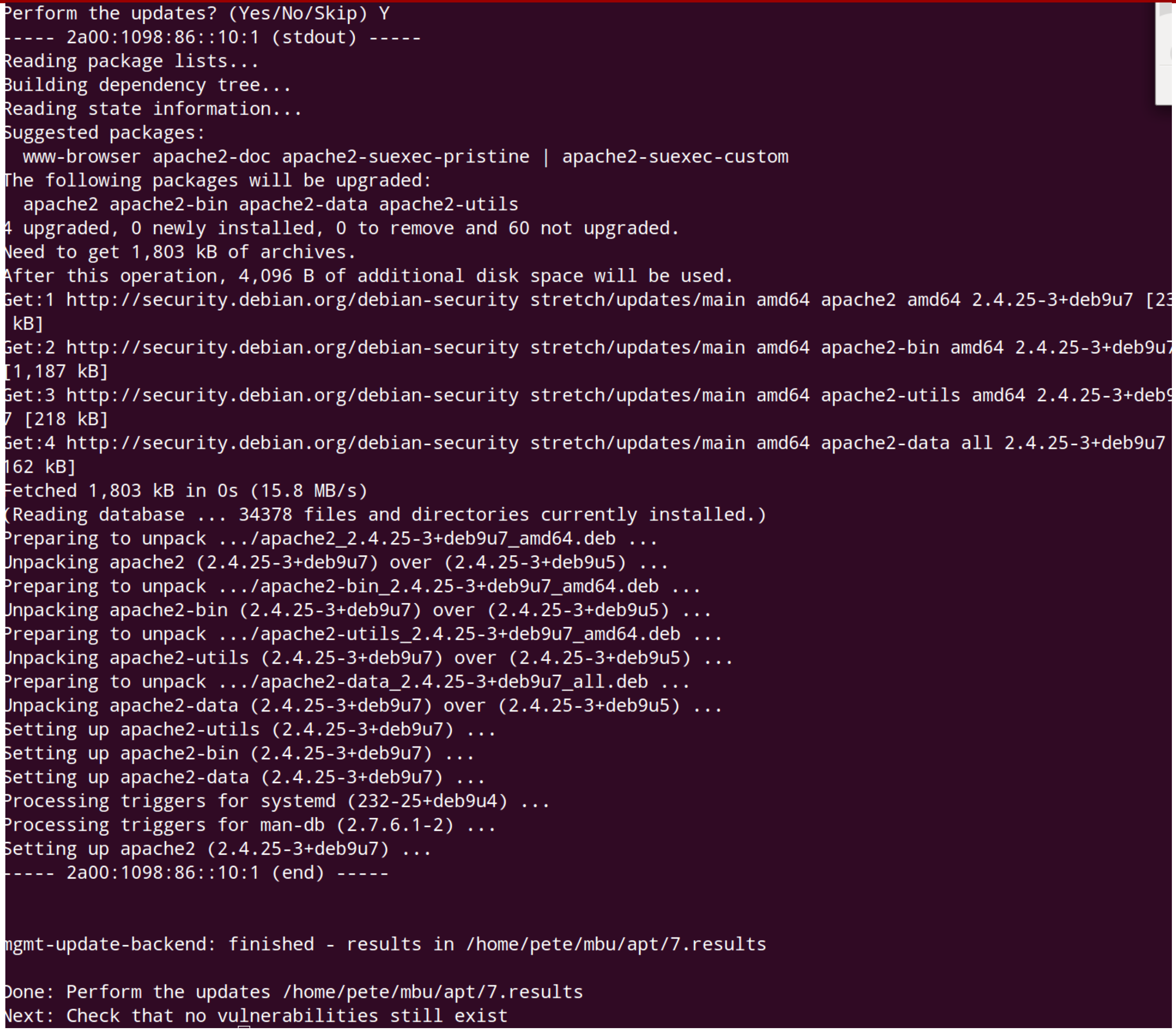
```
pete@admin-gateway:~$ cat mbu/apt/4.template
Subject: Updates to your server

Hi #name,

You are receiving this email because you have a Managed Hosting service
with Mythic Beasts. As part of this service, we install non-urgent
package updates on a monthly basis to maintain the security of your
servers.

The servers concerned are:

#servers

And the packages that will be upgraded this time are:

#packages

A complete list showing exactly what will be upgraded on each server is
appended to this message.

Unless we hear otherwise from you, we will update these packages this
week, starting at:

  07:00 BST (= 06:00 UTC) on Thursday 2019-04-11

The update process should be completed within 1 hour, usually much more
quickly than that. No reboot is required, and any interruption to
service should only last a few seconds as system processes are
restarted.

Regards,

The Security Team
--
Mythic Beasts
support@mythic-beasts.com

///Starts///
#full
///Ends///
```

Managed bulk update process



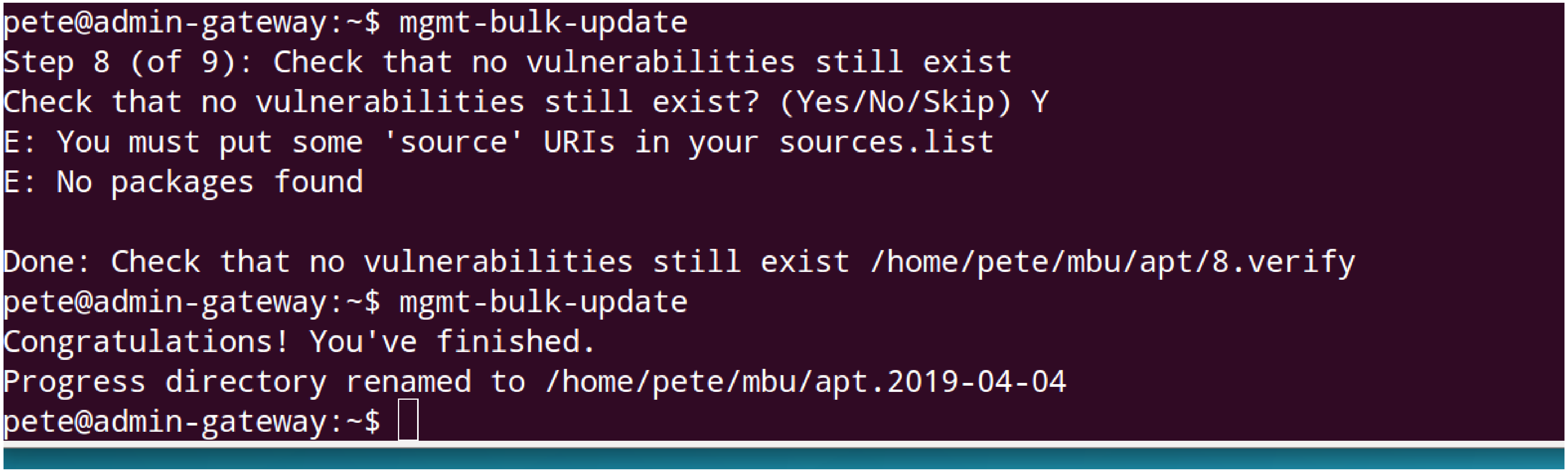
```
pete@admin-gateway:~$ mgmt-bulk-update
Step 5 (of 9): Create the email messages
Creating emails: 2627.. done

Done: Create the email messages /home/pete/mbu/apt/5.emails
Next: Send the email messages
pete@admin-gateway:~$ mgmt-bulk-update
Step 6 (of 9): Send the email messages
Send the email messages? (Yes/No/Skip) N
mgmt-bulk-update: aborting at /usr/local/lib/management/mgmt-bulk-update line 424, <STDIN> line 1.
pete@admin-gateway:~$ mgmt-bulk-update
Step 6 (of 9): Send the email messages
Send the email messages? (Yes/No/Skip) Y
Sending emails: 2627.txt.. email-send: INFO: send: send email to “"Mythic" <admin@mythic-beasts.com>” subject
“Updates to your server”
done

Done: Send the email messages /home/pete/mbu/apt/6.sent
Next: Perform the updates
```

Managed bulk update process



```
Perform the updates? (Yes/No/Skip) Y
----- 2a00:1098:86::10:1 (stdout) -----
Reading package lists...
Building dependency tree...
Reading state information...
Suggested packages:
  www-browser apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following packages will be upgraded:
  apache2 apache2-bin apache2-data apache2-utils
4 upgraded, 0 newly installed, 0 to remove and 60 not upgraded.
Need to get 1,803 kB of archives.
After this operation, 4,096 B of additional disk space will be used.
Get:1 http://security.debian.org/debian-security stretch/updates/main amd64 apache2 amd64 2.4.25-3+deb9u7 [23
 kB]
Get:2 http://security.debian.org/debian-security stretch/updates/main amd64 apache2-bin amd64 2.4.25-3+deb9u7
[1,187 kB]
Get:3 http://security.debian.org/debian-security stretch/updates/main amd64 apache2-utils amd64 2.4.25-3+deb9
7 [218 kB]
Get:4 http://security.debian.org/debian-security stretch/updates/main amd64 apache2-data all 2.4.25-3+deb9u7
162 kB]
Fetched 1,803 kB in 0s (15.8 MB/s)
(Reading database ... 34378 files and directories currently installed.)
Preparing to unpack .../apache2_2.4.25-3+deb9u7_amd64.deb ...
Unpacking apache2 (2.4.25-3+deb9u7) over (2.4.25-3+deb9u5) ...
Preparing to unpack .../apache2-bin_2.4.25-3+deb9u7_amd64.deb ...
Unpacking apache2-bin (2.4.25-3+deb9u7) over (2.4.25-3+deb9u5) ...
Preparing to unpack .../apache2-utils_2.4.25-3+deb9u7_amd64.deb ...
Unpacking apache2-utils (2.4.25-3+deb9u7) over (2.4.25-3+deb9u5) ...
Preparing to unpack .../apache2-data_2.4.25-3+deb9u7_all.deb ...
Unpacking apache2-data (2.4.25-3+deb9u7) over (2.4.25-3+deb9u5) ...
Setting up apache2-utils (2.4.25-3+deb9u7) ...
Setting up apache2-bin (2.4.25-3+deb9u7) ...
Setting up apache2-data (2.4.25-3+deb9u7) ...
Processing triggers for systemd (232-25+deb9u4) ...
Processing triggers for man-db (2.7.6.1-2) ...
Setting up apache2 (2.4.25-3+deb9u7) ...
----- 2a00:1098:86::10:1 (end) -----


mgmt-update-backend: finished - results in /home/pete/mbu/apt/7.results

Done: Perform the updates /home/pete/mbu/apt/7.results
Next: Check that no vulnerabilities still exist
```

Managed bulk update process

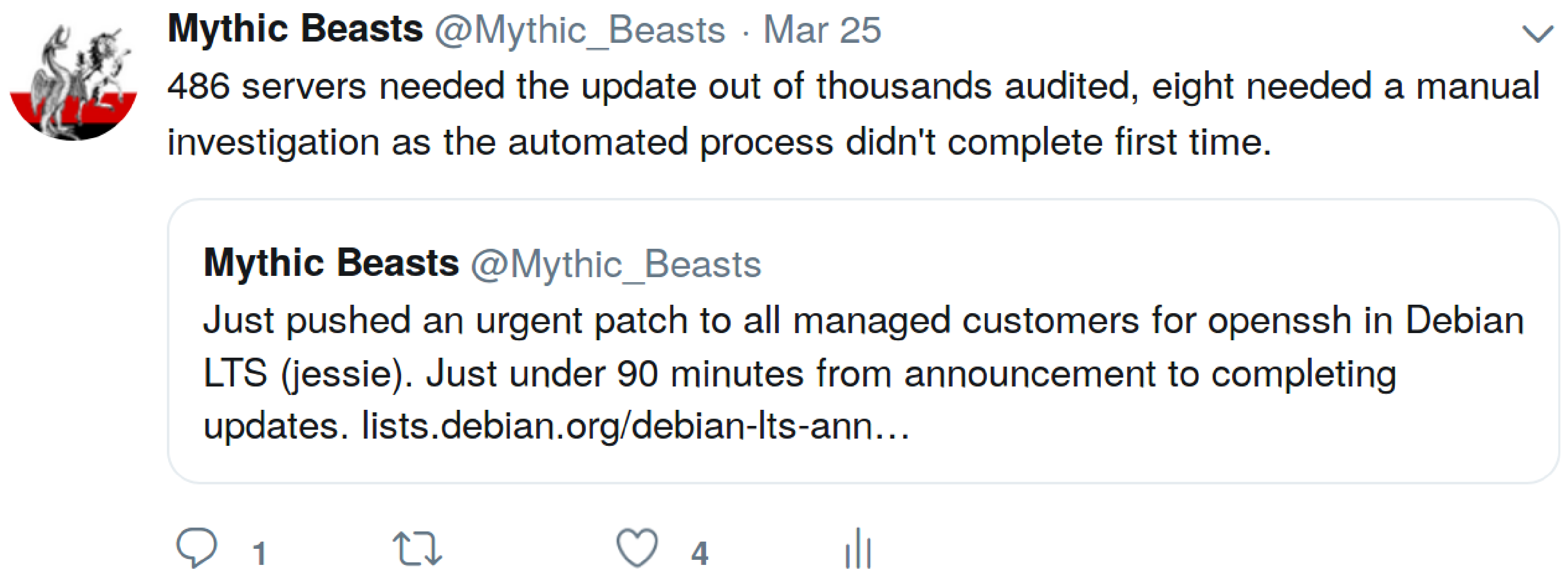


```
pete@admin-gateway:~$ mgmt-bulk-update
Step 8 (of 9): Check that no vulnerabilities still exist
Check that no vulnerabilities still exist? (Yes/No/Skip) Y
E: You must put some 'source' URIs in your sources.list
E: No packages found

Done: Check that no vulnerabilities still exist /home/pete/mbu/apt/8.verify
pete@admin-gateway:~$ mgmt-bulk-update
Congratulations! You've finished.
Progress directory renamed to /home/pete/mbu/apt.2019-04-04
pete@admin-gateway:~$
```

Managed bulk update process

- A 0 day update completes in around an hour
- A monthly update for a data centre completes in about 90 minutes
- We limit updates to 40 at a time
  - Potentially multiple virtual servers on the same physical host
  - Avoid overloading the CPU or IO subsystem

In practice



Mythic Beasts @Mythic_Beasts · Mar 25

486 servers needed the update out of thousands audited, eight needed a manual investigation as the automated process didn't complete first time.

Mythic Beasts @Mythic_Beasts

Just pushed an urgent patch to all managed customers for openssh in Debian LTS (jessie). Just under 90 minutes from announcement to completing updates. lists.debian.org/debian-lts-ann…

1 4

Other systems

- Domains – automate nominet / OpenSRS
- DNS – automate bind
- DNSSEC – automate bind and OpenSRS and nominet
- Hosting accounts – automate apache/dovecot/exim
- Virtual machines – automate KVM on Debian
- Containers – automate LXC on Debian (no docker – we care about persistence!)
- Reporter – daily read firewall / disks / processes etc.
  - Things an experienced admin asks an undocumented machine, we log for every managed server every day.

Other systems

- Same building blocks:
  - SSH for communicating between machines, authentication, encryption all in one go.
  - Perl (mostly, some python, go, haskell)
  - Postgres – We like our data and we want to keep it
  - Create packages for our tools, so they install with apt / yum
  - Our tools get updated by the same process as managed updates

Process (ideal)

- Work out how to do it once and document it
- Turn into a procedure to standardise
- Automate auditing your instances
- Start automating the easiest setup stages
- Repeat until it’s just a handful of commands
- Build into our control panel / operating system package
- Forget how it works
- Discover you have thousands of customers using it
  - and paying for it!

Find out more

https://blog.mythic-beasts.com

@Mythic_Beasts

pete@mythic-beasts.com