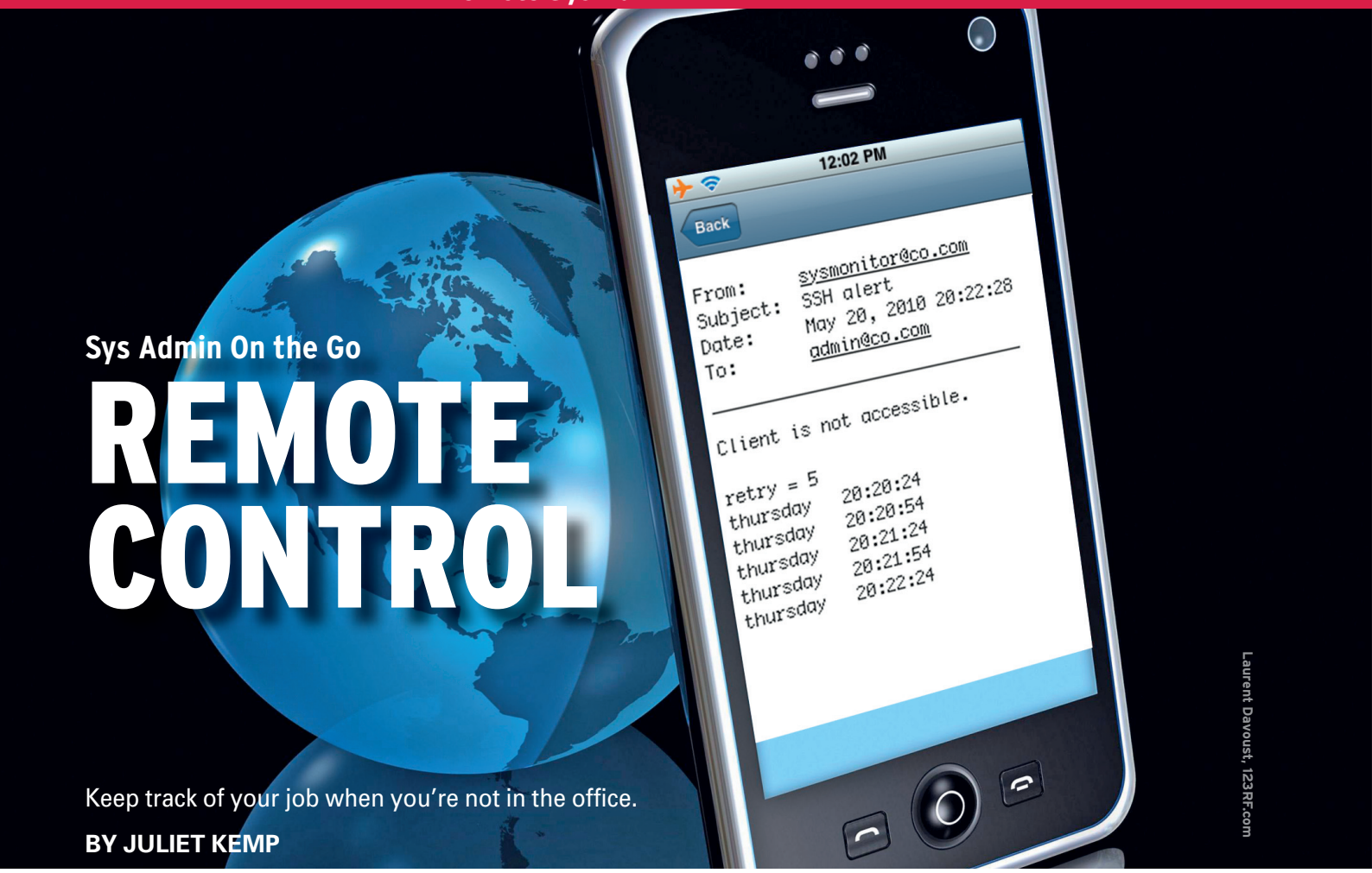

Laurent Davoust, 123RF.com

Sys Admin On the Go

REMOTE CONTROL

Keep track of your job when you're not in the office.

BY JULIET KEMP

As a sysadmin, you don't want to be chained to your desk – it's great to be able to work from home once in a while, or to be able to head off to conferences. But neither do you want the machines you're responsible for to be left unsupervised. Even if you don't mind always being physically in your office during the daytime, there's always the possibility of out-of-hours issues – or what if you're fixing something else at the other end of the building when one of your main servers starts having issues?

In this article, I look at ways to free you from your office by setting up mobile alerts to let you know when something goes wrong, the software you can use with a smartphone to fix it, and other options you have to control your machines from a distance.

Monitoring and Alerts

Several software options will let you keep an eye on your servers and send alerts when something goes wrong. If you set up your monitoring software to send email and you have a smartphone or a similar email-equipped device, you can be warned about problems even when you're not in the office.

Here, I'll do a quick rundown on how to set up email alerts on two of the most popular open source Linux system monitors: Nagios and Hobbit. (I'll assume you already have a working monitoring setup, in that covering the basics of either Nagios or Hobbit is outside the scope of this article.)

Nagios

First, set up an *admins* contact group in the *contacts_nagios2.cfg* file. Defining your admin contacts here means that you can simply refer to the *admin* group elsewhere in your config files. Then, if someone leaves the group, you only have to change the information in a single place. Make sure that a user with your email address is included in this group.

Second, decide which services you want to receive email from. The simplest option is to receive email from all services, by editing the generic service definition (which is used as the basis for all specific services) at *conf.d/generic-service_nagios2.cfg*. Add the code in Listing 1 to the service definition.

The notification interval (in minutes) defines how often you are reminded (here, it's every 24 hours). The *check_period* defines when the service is expected to run (all the time). Time periods such as *24x7* are defined in *conf.d/timeperiods_nagios2.cfg*. The *normal_check_interval* and *retry_check_interval* are in minutes: The service here is set to be checked every five minutes, but if the check is unsuccessful and a retry is made, the retry should happen every minute. Ten retry attempts will be made before something is considered wrong with the service. Although you can reduce this number, if you reduce it too far, you might start getting false positives.

The *notification_period* sets when alerts should be sent – this setup provides for 24/7 notification – and *notification_options* sets when you should receive an alert. For hosts, use *d* to notify on *DOWN* states, *u* to notify on *UNREACHABLE* states, *r* to notify on host recoveries, and *f* to notify when the host starts and stops flapping. For services, use *w* to notify on *WARNING* states, *u* for *UNKNOWN* states, *c* for *CRITICAL* states, *r* for *RECOVERY*, and *f* for start/stop of flapping (when a service or host keeps rapidly changing state, perhaps indicating a problem). Finally, *contact_groups* defines who to contact when a

Listing 1: Notification Setup

```
01 notification_interval          1440
02 is_volatile                    0
03 check_period                   24x7
04 normal_check_interval          5
05 retry_check_interval           1
06 max_check_attempts             10
07 notification_period            24x7
08 notification_options           c,r
09 contact_groups                 admins
```

notification is required (the *admins* group here).

Once you have all that in place, reload Nagios and try turning off SSH on one of the machines being monitored: You should receive a message to the address you set in the contacts file, telling you that the client is not SSH accessible. When you turn it back on, you'll get another alert telling you it's okay again.

One problem you might run into that will prevent an email being sent is a default *From:* line in the email alerts of *nagios*. If your mail server requires a registered address before it will send, mail from this user will be rejected (unless you have a *nagios@mydomain.com* email address set up). If you're using *exim4*, you need to set the *untrusted user* option and then add the line

```
-- -f arealaddress@mydomain.com
```

to the end of the *host-notify-by-email* and *notify-by-email* commands in *commands.cfg*.

If you want different people to be notified at different times (perhaps one person handles evenings and another handles weekends), the best way is to set it up in their *contact* definition in *contacts_nagios2.cfg*. Listing 2 shows a definition that uses my personal email for weekend emergencies when I don't check my work email.

Because this references the time period *weekend*, I also need to set that up

Listing 2: Definition for Weekend Notification

```
01 define contact {
02     contact_name    juliet-personal
03     .....
04     host_notification_period
         weekend
05     service_notification_period
         weekend
06 }
```

in *timeperiods_nagios2.cfg* (Listing 3). The *timeperiod* definition can also handle references to other defined time periods and can exclude times, as well as include times. With this setup, I can now add *juliet_personal* to the *admin* group. However, that contact will only be used during the defined time periods, so I won't get mail to my personal account on weekdays.

Listing 3: Creating a "weekend" Time Period

```
01 define timeperiod {
02     name    weekend
03     timeperiod_name    weekend
04     friday          18:00-24:00
05     saturday        00:00-24:00
06     sunday          00:00-24:00
07     monday          00:00-09:00
08 }
```

Hobbit

To set up email alerts with Hobbit, edit */etc/hobbit/hobbit-alerts.cfg*. Alerts are highly configurable and are set up to define, first, the condition in which an alert should be sent and, second, the action to take for that alert. The code in Listing 4 emails you if any machine (note that the regexp must start with %) is unavailable for 30 minutes (*DURATION*), repeats every 24 hours (*1400* minutes) thereafter, and emails you on recovery (*RECOVERED*). Note that most of this is set up in the *$MAILADMIN* variable, which is used for settings that you're unlikely to change, making for less typing. The settings can be overridden in the line where the variable is used.

As with Nagios, you can set more than one alert for slightly separate conditions. For example, you might want to email the on-call address, as in Listing 5, rather than your daytime work address if the service goes down out of hours.

The important part there is the *TIME* setting: If this is unspecified, the default is "at any time." Here, the second alert is set to be sent only between 6pm (*1800*) and 8am (*0800*) on any day (the first *; use *W* for weekdays, or specify particular days with *0*, Sunday, through *6*, Saturday).

Instead, different people might need to be notified in different situations, in which case *SERVICE* can be specified in the *MAIL* line rather than in the first condition line, as in Listing 6.

Here, the repeat would be every hour, and the admin user will be emailed for problems with the disk or SSH services, with the webmaster being emailed for HTTP problems.

Hobbit also supports the use of the *SCRIPT* keyword rather than *MAIL*, which will run any script you care to plug into it if straightforward email doesn't suit your purposes.

Many other pieces of monitoring software are available, and most, if not all of them, should have similar capabilities for sending out alerts (see the "Sending SMS Messages" box). Nagios and Hobbit are particularly good for larger networks, though.

Software for Mobile Devices

The next step is not just knowing what's going on, but being able to fix it even if you're not there at the right moment. Happily, these days, smartphones are increasingly full of features, and you can do a lot anywhere you have a data connection.

Listing 4: "Host Unavailable" Alert with Hobbit

```
01 $MAILADMIN=MAIL admin@example.com REPEAT=1440 RECOVERED
02
03 HOST=%.* SERVICE=conn
04          $MAILADMIN DURATION>30
```

Listing 5: Setting Multiple Alerts for a Single Service

```
01 $MAILADMIN=MAIL admin@example.com REPEAT=1440 RECOVERED
02 $ONCALLADMIN=MAIL on-call@example.com REPEAT=1440
   RECOVERED TIME=*:1800:0800
03
04 HOST=%.* SERVICE=ssh
05          $MAILADMIN DURATION>30
06          $ONCALLADMIN DURATION>30
```

Listing 6: Alerts for Multiple Services

```
01 $SSHADMIN=MAIL admin@example.com SERVICE=disk,ssh RECOVERED
02 $WEBADMIN=MAIL webmaster@example.com SERVICE=http RECOVERED
03
04 HOST=webserver.example.com
05          $SSHADMIN DURATION>30 REPEAT=1h
06          $WEBADMIN DURATION>30 REPEAT=1h
```

One of the most useful pieces of software the roving sys admin can have is SSH. A variety of SSH is available for most smartphones, including the following:

- For Android, try ConnectBot, found on its Google page [3] or through the Market app. It includes support for SSH keys, which is useful on a mobile platform that might need to reconnect occasionally. Send an Esc by hitting the direction button twice and a Ctrl by hitting it once.
- If you're an iPhone user, try out iSSH. It provides a terminal emulator as well as an SSH client and will also do VNC connections. Private key management is available, and non-standard keys and key combinations are also configurable (it supports the Ctrl and function keys), making Emacs and Vim, for example, usable via iSSH! Multiple connections and configurations can be stored. I find scrolling around the screen slightly irritating, but it is usable. Other alternatives are TouchTerm and pTerm, but neither supports cut and paste or simultaneous connections at present.
- iPad users can use iSSH as well. It provides full-screen iPad support, although a few minor bugs still exist (check their website for details).
- For the PalmOS, pssh provides SSH2 for OS 5 [4], and TuSSH [5] is an alternative if you want SSH1 for Palm OS 4 or 5.

  Not that pssh warns that it might not be entirely secure and shouldn't be used for security-critical applications – in part, because it doesn't use device-specific random number generation. However, it has a neat on-screen keyboard and can support SSH key auth.
- For the Blackberry, MidpSSH is a Java-based client (so it should also work on other Java-compliant devices) that supports a predictive text option, which might be useful if you have a device that doesn't have a full keyboard.

  Also, it supports public key auth, but with no facility for a passphrase for the key, making that a little less useful. MidpSSH does have macro support to make typing long or common strings easier and has a useful documentation blog.
- Symbian has the well-known open source SSH client PuTTY. It supports public key authentication but only for keys created from PuTTYGen in Windows. Excellent documentation is provided either with the download or online.

Sending SMS Messages

Another option, if you don't want to check your email regularly on your smartphone (or don't have a smartphone), is to send regular SMS messages instead [1]. In theory, many providers [2] have an email-to-SMS gateway for phones (i.e., you can email a particular address, and the email will be sent as an SMS to your mobile phone). However, in practice, I had trouble getting this to run on my UK phone.

If a test doesn't work, trying contacting your provider, who might require activation of this feature. (In some cases, activation could take some time.)

Also, you can use the *gsm-utils* or *gnokii* packages (Debian/Ubuntu) to control a GSM (2G) mobile phone plugged into a USB port – and send SMS this way as well. Then, set up an alert script for either Nagios or Hobbit to send alerts via SMS.

This method relies on having a dedicated mobile phone (that supports the appropriate commands – not all phones work with *gsm-utils*) plugged in to your server.

Also, that mobile phone has to have a SIM card and an account with a service provider.

Although I have personally used pssh, ConnectBot, and iSSH on the iPhone, I have not used the others because I don't have access to the appropriate devices.

Additionally, you can use your smartphone to connect to a server via VNC, if your server supports that. In most cases, the SSH command line will be more responsive than a VNC connection, and for most server problems, that should be all you need.

However, for a small class of problems (and especially if you run any, usually Java-based, services that have a graphical admin interface), the VNC option can be useful. iSSH provides a pretty good VNC client, or for Android, you can use *android-vnc-viewer* [6], although this is reasonably new software.

Emailing Your Server

You can also set up your server to respond to specific email. Once, I had a couple of machines that would regularly (because of a specific, and sadly non-resolvable, closed source software problem) lock up their screens.

The solution was to restart GDM. The following line shows an email address, set up in */etc/aliases*, that does this for a particular machine:

```
gdm-restart: "|/usr/sbin/restart-gdm"
```

This pipes the incoming email into the named script, which can do a quick-and-dirty security check for the sender of the email, as shown in Listing 7.

Two final authorization changes are needed to make this work. First, if you're running *exim4*, you need to au-

Listing 7: */usr/sbin/restart-gdm* Script

```
01 #!/usr/bin/perl -w
02
03 use strict;
04
05 my $legit_sender = "juliet\@
   mydomain.com";
06
07 while(<>) {
08    if ( /^From:/ ) {
09       `sudo /etc/init.d/gdm
   restart` if ( /$legit_sender/ );
10        exit 0;
11    }
12 }
```

thorize that scripts are run from email aliases by editing */etc/exim4/exim4.conf.template* to include this line (this should be outside the *SYSTEM_ALIASES_PIPE_TRANSPORT* ifdef block),

```
pipe_transport = address_pipe
```

then reload it (*/etc/init.d/exim4 reload*) for the change to take effect. Other mailers might have to do something similar.

Second, you'll need to give the exim user (*Debian_exim* in Debian Lenny) sudo rights to run this particular command without a password. Run *visudo* and add this line at the bottom:

```
Debian-exim machinename = NOPASSWD: /etc/init.d/gdm
```

Now send mail (content doesn't matter here) to *gdm-restart@mymachine.mydomain.com*, and GDM will be restarted!

IMPORTANT NOTE! This script is NOT particularly secure! A straightforward spoof of the *From* line in an email is possible. Reloading GDM isn't, as a rule, a particularly dangerous thing to do (although it could seriously irritate a user who was in the middle of something), but it's important to be aware that restarting software does carry a risk of an attacker being able to run a replacement script or application if they've managed to insert one into your system. Be very careful what you allow this sort of script to do, and use this kind of automation with caution.

INFO

[1] Email to SMS: *http://www.mutube.com/projects/open-email-to-sms/*

[2] Email gateways: *http://www.mutube.com/projects/open-email-to-sms/gateway-list/*

[3] ConnectBot: *http://code.google.com/p/connectbot/*

[4] pssh – SSH2 for Palm OS 5: *http://www.sealiesoftware.com/pssh/*

[5] TuSSH – SSH1 for Palm OS 4 and 5: *http://www.tussh.com/*

[6] Android VNC viewer: *http://code.google.com/p/android-vnc-viewer/*

However, used carefully this technique can be really useful. Another possibility, for example, is to write a script that acknowledges a Nagios alert so that it doesn't keep bugging you if you've already decided that it's not urgent.

Conclusion

The scope for remote manipulation of your servers is extensive – as is their ability to manipulate you remotely by sending relevant information to you wherever you are! By trying out some of these techniques, you could free up your working life a little.

Just remember to mention it in your performance review if, as I did once, you end up fixing a web server from a tent in the middle of a field somewhere! ■

THE AUTHOR

Juliet Kemp has been playing around with Linux ever since she found out that it was more fun than Finals revision and has been a sys admin for around 10 years. She is the author of *Linux Systems Administration Recipes: A Problem-Solution Approach.*