

How to monitor your system from your mobile



Fancy a day at the beach, but worried about your server going down? Dave Coulson sets you up with SMS

Every day the world just seems to become that bit smaller. You find yourself running software downloaded from a site in Australia on your server located across the Atlantic.

While this is usually an advantage, floating around between different places without access to your usual tools for prolonged periods of time isn't ideal, especially if you're running mission critical applications.

There is, however, one handy piece of hardware which always comes in useful: the trusty cellular phone. Wherever you are – sitting in an airport lounge, the back of a NYC cab, or even in your kitchen – it's invaluable. It's only a shame that your Web server can't give you a call when it feels a little under the weather.

But hold on, it can. Use an e-mail to Short

Message Service (SMS) gateway and your machine can contact you wherever in the world your mobile phone works. You can get it to send you the latest bandwidth statistics, security problems or

even the latest stock quotes. You can be up to the second with how much you've lost in the latest dot-com IPO.

Monitor anything

Although we're just going to deal with system monitoring here, once the basics are up and running it can easily handle more specific jobs.

First, we'll assume that both the monitoring machine and the SMTP server are totally stable. In a perfect world we'd have redundancy. The next thing you need to decide is what exactly you want to monitor and how frequently you want to monitor it.

A Web server should probably be checked every five or ten minutes, but an SMTP server is safe for an hour or two as relays provide automatic backup. It's always handy to check the machine is actually there so you know if it's simply a software problem, or if it's something nastier like a physical networking issue.

There are a number of system monitoring tools available for Unix machines. Mon (bet you can't guess what that's short for!) is one of the more documented ones around and comes with a huge number of example scripts and configuration files. You can monitor pretty much every service you can think of quickly and easily.

The general method behind Mon is that you specify a list of machines you want to monitor, which services, plus a number of alert timing options. You also specify a command which should be run when it encounters a problem, such as a mailer or a script which takes that particular machine out of rotation.

There are a number of levels of checking. If you had a Web server you could check to see if the machine was actually there by pinging it. You could see if the Web server was up by trying to open a connection to port 80 and finally to check it's actually working by trying to fetch a known URL.

As well as the various types of monitors and alerts, you can also specify a time period in which the monitoring takes place. You might have it perform an automated task during the night or when an administrator is unavailable.

Once you've decided upon your general monitoring strategy, scripting a mon.cf script to perform the monitoring is easy. If you wanted to have your Web server checked

floating around between different places **without access** to your usual tools for prolonged periods of time **isn't ideal**

HARDCORE SMS

News for nerds

How to get the stuff that matters while on the go

You can send pretty much anything to a mobile phone, as long as it's under 160 characters and isn't too difficult to read on a small screen. Ideal examples of this are news headlines, pulled from a Web site. And what better site than www.slashdot.org. There are loads of scripts out there for grabbing headlines, but we'd recommend

SlashWatch (www.tappy.com/slashwatch/). It pulls slashdot.org headlines, stores them in a mysql database then e-mails them out to a list of users.

Freshmeat.Net is also useful if you want a particular program release sent to your phone, but you'll have to throw a good filter on it – there are far too many additions in a day to have everything sent.



Mobile notification doesn't have to be for business use. It's handy for those of us who consider Slashdot.Org a way of life.

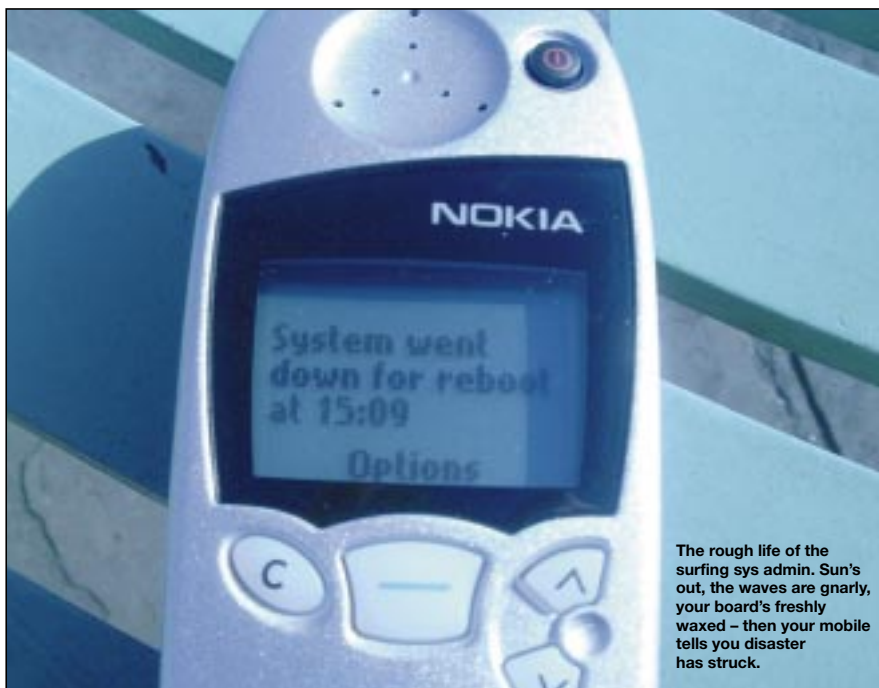


Now you'll never need to go into the office again. Well, until that **dire message** pops up on your phone

every five minutes you would use an entry similar to the following:

```
hostgroup wwwservers www
watch wwwservers
service http
interval 5m
monitor http.monitor
period wd {Mon-Fri} hr {7am-10pm}
alert mail.alert -S "Web
Server 'www' is down"
bofh@domain.com
alertevery 45m
period wd {Sun-Sat}
alert mail.alert bofh
sms@domain.com
alertevery 6h
```

One thing to take note of here is the 'alertevery' parameter. After the first alert, you'll only receive notification of it being



The rough life of the surfing sys admin. Sun's out, the waves are gnarly, your board's freshly waxed – then your mobile tells you disaster has struck.

down every 45 minutes. The next section will alert you ever six hours, via your SMS gateway. You'll have noticed that we've sent the SMS e-mail to our local mail server, rather than straight to our SMS gateway. Sending it through a local mail server first enables you to easily filter out e-mail which has been sent to your SMS e-mail account, so you don't get spam or other junk sent to your mobile phone. Once it's gone through the

filters, it's simple enough to bounce it to <username>@sms.genie.co.uk or to another SMS service.

Now you'll never need to go into the office again. Well, until that dire message pops up on your phone. **PCP**

M.I.A

Chris Jones is unwell...

STEP-BY-STEP

How to configure your system

How to send notifications of problems via SMS

1 Before you can do anything with your systems, you need to set up a SMS gateway so you can send e-mails to your phone. One of the more popular ones is Genie (www.genie.co.uk) run by BT Cellnet. Although it's a free service, you'll pay to receive the text messages at the price set by your operator. Once you've registered with Genie, you can test it by sending a couple of test messages to <username>@sms.genie.co.uk.

2 Next, head over to FreshMeat.net and download 'Mon'. You'll need to download two separate tar balls, one containing the actual Mon program and the other having all the required Perl modules for the various scripts. If you prefer to use 'cpan', then the module names are given in the INSTALL document. Each of the modules has its own test script, so it's a good idea to use it.

3 Once you've install the Perl modules, you need to build the monitoring binaries. Simply 'cd mod-0.38.15/mon.d' and run 'make'. You'll then have to add a couple of lines to /etc/services:

```
mon 2583/tcp # MON
mon 2583/udp # MON traps
```

It's then simply an issue of customising the default example.cf file to suit your needs. The following example would simply watch the http running on 'localhost':

```
cfbasedir = /usr/lib/mon/etc
alertdir = /usr/lib/mon/alert.d
mondir = /usr/lib/mon/mon.d
```

```
maxprocs = 20
histlength = 100
randstart = 60s
authtype = getpwnam

hostgroup wwwservers 127.0.0.1

watch wwwservers
service http
interval 4m
monitor http.monitor
period wd {Sun-Sat}
alert mail.alert bofh@domain.com
alertevery 45m
```

4 Running 'mon' is done by running the following from the source tree:

```
./mon -f -c etc/mon.cf -s mon.d -a alert.d
```

'etc/mon.cf' is the path to your mon configuration file. If you miss out the '-c etc/mon.cf' it looks in /etc/mon/mon.cf for the file. Once you've got it up and running happily, you can check its status by running:

```
./clients/moncmd -s localhost list pids
```

Once you've established that it's working properly, you can copy the binaries into the relevant directory (/usr/sbin usually) and the alert.d, mon.d and the .cf files to /usr/lib/mon/.