

How to use FoxBox for sending SMS from Nagios

{jfalternative}44|content|There are no translations available{/jfalternative}OFFICIAL NAGIOS INTEGRATION METHOD for ACME SYSTEMS SMS FoxBox (AVAILABLE ON WIKI)

From NagiosCommunity

Contents [hide] 1 Overview 1.1 SMS FoxBox 2 Initial Setup 2.1 SMS FoxBox Unit:2.2 Monitoring Server: 3 sendSMS.sh Script4 Monitoring Configuration5 Additional Notes if (window.showTocToggle) { var tocShowText = "show"; var tocHideText = "hide"; showTocToggle(); } Overview SMS FoxBox

The SMS FoxBox is a low cost SMS/MMS Gateway. It uses a tiny embeded linux system with a GMS/GPRS modem. You just need to supply the SIM card.

Info from their site:

- Up to 30 incoming SMS/min with a common SIM card
- 100% solid state hardware MMS / SMS gateway in just 105x110x45mm
- Fully featured Web interface for MMS and SMS management
- Email To SMS and SMS to Email gateway
- Fully customizable software with PHP, C and Bash coding
- Quad band GSM modem
- Linux embedded OS 2.6.x
- Apache embedded web server
- SQLite embedded SQL server
- Expandable set of gateway functions

Initial Setup SMS FoxBox Unit:

- We installed a SIM card into the SMS FoxBox by following the included documentation.
- The SMS FoxBox was installed on our network and given an IP address of 192.168.1.98
- Verified connection to device by browsing to 192.168.1.98 and using the default credentials of 'Admin' with password 'GsmBox2006!'. Monitoring Server:
- Nagios 3.0.6 was compiled and installed on the monitoring server
- Version 1.4.13 of the Nagios plugins were installed on the monitoring server.
- A custom script was installed and titled 'sendSMS.sh' sendSMS.sh Script

We created a script named 'sendSMS.sh' and placed it in /usr/local/nagios/libexec/ directory: The script is shown below:

```
#!/bin/bash
#
# For use with SMS Fox Box. This is a basic script that calls the 'send_sms.php' script on their server
# then parses the result to see if the message was sent or rejected.
#
# This is intended for use with Nagios notifications but could be adapted to other uses.
#
# Note: a positive message using their service means that it was accepted and queued. It is up to the
# SMS Fox Box to process their queue, determine if the SIM Card has credits, or filter it through a
# separate gateway. This method has the potential to overload the SMS Fox Box webserver if used with
# high numbers of Nagios Notifications.
#
#
# Dependencies: bash, curl, grep, sed must be installed and in your system path.
#
# Nagios Enterprises, LLC.
# 12182008 - myyoung@nagios.org
#
## Variables to modify by hand.
HOSTNAME=192.168.1.98
BASIC_AUTH=0 # 1 if basic auth is used; 0 if not.
USERNAME="Admin"
PASSWORD="GsmBox2006!"
#CURL_CONNECTION_TIMEOUT=30 # set the timeout for the connection phase of the curl; in seconds
#CURL_MAX_TIME=40 # set the max length of the curl connection; in seconds
```

Parameter Checking

EXPECTED_ARGS=3 #expected number of arguments

E_BADARGS=65 #exit code with bad arguments

```
if [ $# -ne $EXPECTED_ARGS ]; then
    echo
    echo "Usage: $0 <from addr> <phone number> <message text>."
    echo "<from addr>:          Name or Address of Nagios Server"
    echo "<phone number>:  Phone Number for the location of the sms to be sent"
    echo "<message text>:  The text of the message in quotes. ex. \"message content\""
    echo
    echo "For use with SMS Fox Box. Uses curl to call foxbox \'send_sms.php\' script,"
    echo "parses the result, returns 0 for positive, 1 for negative, and 65 for error"
    echo
    echo "12182008 - myyoung@nagios.org"
    echo
    exit $E_BADARGS
fi
```

```
## Initialize variables.. Should not need to modify.
FROMADDR=$1
PHONENUMBER=$2
MESSAGE_TEXT=$3
REMOTE_URL="" # Will be set after determining Basic Authentication usage.
```

```
## Check to see if we should use Basic Authentication.
if [ "$BASIC_AUTH" = "1" ]; then
    REMOTE_URL=http://$USERNAME:$PASSWORD@$HOSTNAME/source/send_sms.php
else
    REMOTE_URL=http://$HOSTNAME/source/send_sms.php
fi
```

```
## Send variables to url via POST; grep and sed result to parse 1 for confirmed positive, 0 for negative, nothing for error.
PARSED_RESULT=`curl -F "from=$FROMADDR" -F "nphone=$PHONENUMBER" -F "testo=$MESSAGE_TEXT" -F "send=send" -F "nc=" $REMOTE_URL | grep "p class" | sed -e "s/.*/1/" -e "s/.*/0/"`
```

```
# Case statement. 1 for confirmed, 2 for negative, default case for error.
case "$PARSED_RESULT" in
    1 )
        #echo "positive"
        exit 0
        ;;
    2 )
        #echo "negative"
        exit 1
        ;;
    * )
        #echo "unexpected text or error"
        exit 65
        ;;
esac
```

Monitoring Configuration

We modified our Nagios configuration files to include new notification scripts: # 'notify-host-by-foxbox' command definition

```
define command{
    command_name  notify-host-by-foxbox
    command_line  /usr/local/nagios/libexec/sendSMS.sh Nagios $CONTACTPAGER$ "Host Alert: $HOSTNAME$\nHost State: $HOSTSTATE$\nDate/Time: $LONGDATETIME$"
}
```

'notify-service-by-foxbox' command definition

```
define command{
    command_name  notify-service-by-foxbox
    command_line  /usr/local/nagios/libexec/sendSMS.sh Nagios $CONTACTPAGER$ "Service Alert:
```

```
$HOSTALIASS/$SERVICEDESC$\nService State: $SERVICESTATE$\nDate/Time: $LONGDATETIMES$"
```

Note: SMS messages in this script will be sent out to the contacts pager number.

We then added a new contact and contact group using the new notification script and having a valid phone number:

```
define contact{
    contact_name test-contact
    use generic-contact
    alias tester
    email valid@domain.tld
    host_notification_commands notify-host-by-sendsms
    service_notification_commands notify-service-by-sendsms
    pager 12453683421
}
```

```
define contactgroup{
    contactgroup_name test-group
    alias test-group
    members test-contact
}
```

Now we need to make sure we have host and service setup for notifications with this user, this is normally done broadly using templates, let us define a generic template that hosts and services should use: define host{

```
name generic-host
notifications_enabled 1
event_handler_enabled 1
flap_detection_enabled 1
failure_prediction_enabled 1
process_perf_data 1
retain_status_information 1
retain_nonstatus_information 1
notification_period 24x7
register 0
max_check_attempts 3
check_interval 5
retry_interval 1
check_command check-host-alive
contact_groups test-group
notification_interval 0
notification_options d,u,r,f,s
register 0 ; a template definition
}
```

```
define service{
name generic-service
active_checks_enabled 1
passive_checks_enabled 1
parallelize_check 1
obsess_over_service 1
check_freshness 0
notifications_enabled 1
event_handler_enabled 1
flap_detection_enabled 1
failure_prediction_enabled 1
process_perf_data 1
retain_status_information 1
retain_nonstatus_information 1
is_volatile 0
check_period 24x7
max_check_attempts 3
normal_check_interval 5
retry_check_interval 2
notification_options w,u,c,r,f,s
```

```
notification_interval    0
notification_period      24x7
contact_groups           test-group
register                  0 ; a template definition
}
```

Note: After creating or editing the templates you will then need to make sure that hosts/services are using them. Both hosts and service use the directive 'use <name of template>' inside the brackets of the object definition.

After saving our configuration files, we verified our configuration files and started Nagios: /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg /etc/init.d/nagios restart

Once Nagios was restarted and external commands are enabled, we can force a notification to be sent by clicking on a host/service to open extended information page, then by selecting 'Send custom host notification', and on the next screen selecting 'forced' then 'commit'. Additional Notes

- This document represents one method of sending notifications. That being using the FoxBox web service to POST data to the FoxBox.
- Additionally we tested other methods that FoxBox supports, including using scp to upload files to a spool directory on the device, and by using ACME's email2sms script reported below.

METHOD OF INTEGRATION USING MAIL2SMS GATEWAY FUNCTION (TESTED BY NAGIOS)

Think how nice will be to receive SMS notification from NAGIOS for your monitored services, in this tutorial we will see how to send notifications via SMS.

Using SMS FoxBox EMAIL to SMS gateway function it is really simple, just setup NAGIOS to send out email notifications and then setup SMS FoxBox gateway to forward email to sms followinf this guide:

http://www.smsfoxbbox.it/the_news/latest_news/how_to_join__sms_to_your_server_environment.html

Here there are some sample files to tune up procmail configuration so that the system will divert each message to the designed mobile phone number.

To divert all email messages make sure that the .procmailrc script located inside /etc/ folder looks like this

```
SHELL=/bin/sh
VERBOSE=OFF
MAILDIR=/mnt/flash/root/procmail
DEFAULT=/mnt/flash/spool/mail/mails
LOGFILE=/mnt/flash/spool/mail/procmail.log
#OUTFILE="/mnt/flash/spool/smsgw.out.XXXXXXX"
OUTFILE=$(/bin/mktemp /mnt/flash/spool/outgoing/smsgw.out.XXXXXXX)
```

```
:0
* ^Subject:.
{
:0 b
| grep -v ^$ | sed '/To:/G' | /etc/sms/scripts/email2sms
}
```

Here you can download the script email2sms that you have to put in /etc/sms/scripts/.

Replace in the script the XXXXXXXXXXXX with the designed mobile number and if you want uncomment blocks to add more destinations.

```
#!/bin/sh
#SCRIPT: MAIL2SMS
#AUTHOR: Davide Cantaluppi kanta@kdev.it
#DATE: 28.02.06
#REV: 1.1a
#PLATFORM: Not platform dependent
PURPOSE: This is a "parser" for the message passed from procmail
# print the whole email body as the output to OUTFILE
```

```
cat > /var/temp.txt OUTFILE=$(mktemp /mnt/flash/spool/outgoing/smsgw.out.XXXXXX)
```

```
echo "From: Nagios" > $OUTFILE
echo "To: XXXXXXXXXXXX" >> $OUTFILE
echo "" >> $OUTFILE
cat >>$OUTFILE
#NAGIOS SECOND NOTIFICATION
#OUTFILE=$(mktemp /mnt/flash/spool/outgoing/smsgw.out.XXXXXX)
#echo "From: Nagios" > $OUTFILE
#echo "To: 39XXXXXXXX" >> $OUTFILE
#echo "" >> $OUTFILE
#cat /var/temp.txt >>$OUTFILEEA NEW PRODUCT EASY GUARDIAN CAN OFFER TO YOU NETWORK MONITORING
IN MINUTES
Easy Guardian is a small low cost Linux appliance with a fully Web-based system that p
complete network monitoring solution. Easy Guardian , is a customizable and easy to maintain system for monitoring the
real-time availability of network devices, servers (Windows, Unix, Linux) and all network delivered services in any IT
infrastructure.
```

Easy Guardian displays your system and network status on a color-coded Web page that proactively notifies you of problems immediately via SMS. Easy Guardian is built for System & Network administrators tasked with managing the availability and performance of hundreds or thousands of servers and network devices on a limited budget.

Easy Guardian enables you to monitor any server, any device, on any network within minutes.features:

- Monitor Servers (LINUX, WINDOWS APPLE MACINTOSH...)

- Monitor Networks
- Monitor Desktops
- Monitor Devices
- Monitor Databases
- Monitor Performance
- Log all data and draw charts and polls

Product Tour