



Overview

Nagios Enterprises tested ACME Systems' SMS FoxBox to ascertain the ease and availability of integration with Nagios. This document will summarize the steps we took to successfully integrate the FoxBox unit for use with Nagios notifications.

The SMS FoxBox was utilized successfully in sending sms notifications with Nagios through the use of the included web service.

Components Used In Testing

SMS FoxBox:

Model:	SMS FoxBox
Kernel:	2.6.15
CPU:	ETRAX 100 LX v.2
System Board:	FOX LX832
Carrier Board:	FOX GM862-QUAD GSM/GPRS

Monitoring Server

Operating System:	Fedora Core 8
Nagios Version:	3.0.6

Initial Setup

SMS FoxBox Unit:

1. We installed a SIM card into the SMS FoxBox by following the included documentation.
2. The SMS FoxBox was installed on our network and given an IP address of 192.168.1.98
3. Verified connection to device by browsing to 192.168.1.98 and using the default credentials of 'Admin' with password 'GsmBox2006!'.



Monitoring Server:

1. Nagios 3.0.6 was compiled and installed on the monitoring server
2. Version 1.4.13 of the Nagios plugins were installed on the monitoring server.
4. A custom script was installed and titled 'sendSMS.sh'

sendSMS.sh Script

1. 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 "texto=$MESSAGE_TEXT" -F "send=send" -F
"nc=" $REMOTE_URL | grep "p class" | sed -e "s/.*confpos.*1/" -e "s/.*confneg.*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

2. 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:
$HOSTALIAS$/$SERVICEDESC$\nService State: $SERVICESTATE$\nDate/Time: $LONGDATETIME$"
}
```

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

3. 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
}
```

4. 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.

5. 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
```

6. 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'.

The screenshot shows the Nagios web interface for host 'zeus'. On the left, the 'Host State Information' panel shows the host is 'UP' for 4d 16h 54m 28s. The 'Host Commands' panel lists various actions, with 'Send custom host notification' highlighted in red. To the right, the 'Command Options' form is displayed, with 'Host Name' set to 'zeus', 'Forced' checked, and 'Author' set to 'Nagios Admin'. The 'Commit' button is also highlighted in red. A red warning message at the top reads: 'You are requesting to send a custom host notification'. Below the form, a note states: 'Please enter all required information before committing the command. Required fields are marked in red. Failure to supply all required values will result in an error.'

Note:

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.

Company Contact Information

For more information on monitoring ACME SYSTEMS' SMS FoxBox units with Nagios, contact the following companies:

Nagios Enterprises, LLC
P.O. Box 8154
Saint Paul, MN 55108

Email: inquiries@nagios.com
Web: www.nagios.com
US: (888) 624-4671
Int'l: +1 (651) 204-9102

ACME SYSTEMS srl
Via Nettuno 16/A
00055 Ladispoli (RM) Italy
P.IVA/VAT: IT08114831004

Web: <http://www.acmesystems.it>
FAX: (+39) 06.622.765.31