

NRPE – Installation and Configuration at IPCop¹

v2.1.2² Router (Unofficial Manual) v1.1

Spis treści

1. IPCop.....	2
A. NRPE Addon instalation.....	2
B. NRPE configuration.....	2
a. nrpe.cfg.....	2
b. iptables	3
c. /etc/services	3
C. External plugins for monitoring (cpu/mem/hdd usage, software RAID monitoring).	3
a. check_cpu.sh	5
b. check_mem.pl	5
c. check_raid	6
d. check_disk	6
D. Run NRPE daemon.....	7
2. Nagios Server.....	8
A. Test if everything is working.....	8
a. check_nrpe	8
b. check_nrpe with check_command.....	8
B. Host/Service configuration.....	8
a. Host configuration.....	9
b. Service configuration.....	9
C. RAID notifications	11

¹ <http://en.wikipedia.org/wiki/IPCop>

² <http://www.ipcop.org/download.php>

I hope this tutorial will help you configure monitoring IPCop Router via Nagios's NRPE. I spent much time to figure out how to do this, I asked for help people from <http://support.nagios.com> and this helped me a lot. If you have questions mail me: thor1990.03.15(at)gmail(dot)com – Polish and English preferred.

1. IPCop

A. NRPE Addon installation.

First of all we need to download nrpe addon for IPCop³. You can use also the one I'll give you with this manual⁴. You need to put it to your IPCop Router (eg. via WinSCP) or the way you like. First of all we need to unpack arch:

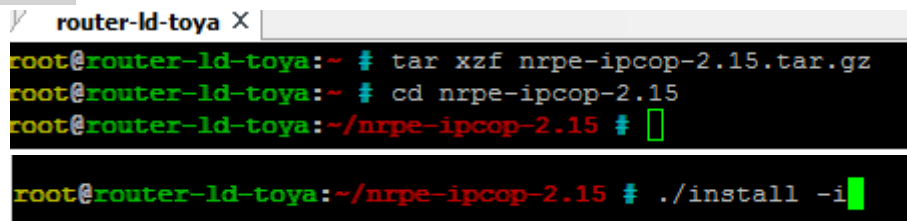
```
tar xzf nrpe-ipc-2.15.tar.gz
```

next we must enter the dir with installation files, type:

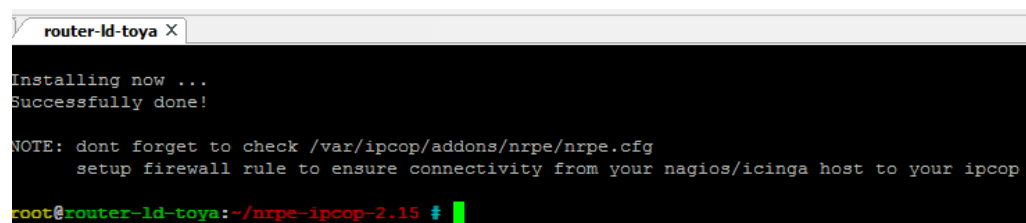
```
cd nrpe-ipc-2.15/
```

the installation process is the simplest as it can be, you need to type only:

```
./install -i5.
```



```
router-ld-toya X
root@router-ld-toya:~ # tar xzf nrpe-ipc-2.15.tar.gz
root@router-ld-toya:~ # cd nrpe-ipc-2.15
root@router-ld-toya:~/nrpe-ipc-2.15 # 
root@router-ld-toya:~/nrpe-ipc-2.15 # ./install -i
```



```
router-ld-toya X
Installing now ...
Successfully done!

NOTE: dont forget to check /var/ipcop/addons/nrpe/nrpe.cfg
      setup firewall rule to ensure connectivity from your nagios/icinga host to your ipcop
root@router-ld-toya:~/nrpe-ipc-2.15 #
```

B. NRPE configuration.

After we install nrpe-ipc plugin we need to do a few things to make it work. All files we can find at `/var/ipcop/addons/nrpe/`

a. nrpe.cfg

type `cd /var/ipcop/addons/nrpe` then edit `nrpe.cfg` file, I prefer vi(m). Type `vi nrpe.cfg`.

`server_port=5666` – default port for Nagios „conversation“ ;)

`server_address=192.168.1.1` – default IPCop has two network cards – RED (public IP from ISP) and GREEN (local network – LAN). In our company we

³ <http://ipcop.stankowicz-development.net/>

⁴ Note that the day I write this manual (31.03.2014) actual version of NRPE plugin was 2.15.

⁵ If we want deinstall it, we need to put: `./install -u`

have VPN configured between our IPCops so I added there our local IP (IP GREEN card like 192.168.1.1, 192.168.2.1, etc.).

`allowed_hosts=127.0.0.1` – we need to put here IP of Nagios' Server otherwise IPCop won't allow for communication.

`Debug=0` – if we put `1` then we enable debug mode – logging

`Command[check_<command-name>]=/place/where/command/file/is` – `arg1` – `arg2`, (etc). – here we can add our commands with plugins for what we want checking.

When we finish, we have to save settings and close `nrpe.cfg` file, just type `:wq` [Return].

b. iptables

We need allow Nagios for connect to our IPCop Router, to do this we have to add one line for `iptables`. Type `vi /etc/rc.d/rc.firewall` and find section `iptables_init()` { go to end of it (before `} iptables_red()` { }) and add:

`#NRPE`

`/sbin/iptables -I INPUT -p tcp -m tcp --dport 5666 -j ACCEPT`

Save and exit this file (`:wq` [Return])

```
#NRPE
/sbin/iptables -I INPUT -p tcp -m tcp --dport 5666 -j ACCEPT
}

iptables_red() {
```

To make changes take effect we can restart whole router (*reboot*) or we can type `/etc/rc.d/rc.firewall restart`

c. /etc/services

Open file `/etc/services` and make sure you have `nrpe 5666/tcp` port added (it should be added by `nrpe-icop` plugin installer), open it `vi /etc/services` and type `/5666` (/ - search, XZY – value we're looking for).

At the end of file we should have

```
nrpe 5666/tcp
```

C. External plugins for monitoring (cpu/mem/hdd usage, software RAID monitoring).

I'll show how to configure some interesting plugins (I also added them with this procedure). Before we start adding command we need put the plugins for plugins directory and make them executable⁶:

Copy:

```
cp check_cpu.sh /var/ipcop/addons/nrpe/plugins/
cp check_mem.pl /var/ipcop/addons/nrpe/plugins/
```

⁶ I had only these three plugins at `/root` directory so I did faster method (screen below) ;)

```
cp check_raid /var/ipcop/addons/nrpe/plugins/
```

Executable attributes:

```
chmod +x /var/ipcop/addons/nrpe/plugins/check_cpu.sh
```

```
chmod +x /var/ipcop/addons/nrpe/plugins/check_mem.pl
```

```
chmod +x /var/ipcop/addons/nrpe/plugins/check_raid
```

```
root@router-ld-toya:~ # chmod +x check_*
root@router-ld-toya:~ # cp check_* /var/ipcop/addons/nrpe/plugins/
```

check_disk is default plugin added with nrpe-ircop addon so we do not need do these operations for it.

We can run those plugins and check if they're returning good values:

```
cd /var/ipcop/addons/nrpe/plugins/
```

a. check_cpu.sh

when we run `./check_cpu.sh` we'll get error – *bc: command not found*.

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_cpu.sh
./check_cpu.sh: line 118: bc: command not found
./check_cpu.sh: line 119: bc: command not found
./check_cpu.sh: line 120: bc: command not found
./check_cpu.sh: line 121: bc: command not found
./check_cpu.sh: line 122: bc: command not found
./check_cpu.sh: line 123: bc: command not found
./check_cpu.sh: line 125: bc: command not found
./check_cpu.sh: line 126: bc: command not found
./check_cpu.sh: line 129: bc: command not found
./check_cpu.sh: line 131: bc: command not found
./check_cpu.sh: line 133: bc: command not found
./check_cpu.sh: line 135: bc: command not found
./check_cpu.sh: line 138: bc: command not found
./check_cpu.sh: line 139: bc: command not found
./check_cpu.sh: line 142: bc: command not found
./check_cpu.sh: line 144: bc: command not found
./check_cpu.sh: line 146: bc: command not found
OK - user: , nice: , sys: , iowait: , irq: , softirq: idle: | 'user'='nice'='sys'='softirq'='iowait'='irq'='idle'='
```

We need add **bc** to `/usr/bin` and make it executable:

```
cp ~/bc /usr/bin
```

```
chmod +x /usr/bin/bc
```

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # cp ~/bc /usr/bin/
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # chmod +x /usr/bin/bc
```

Lets try again `./check_cpu.sh`

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_cpu.sh
OK - user: 3.30, nice: 0.50, sys: 3.30, iowait: 0.50, irq: 0.50, softirq: 0.50 idle: 94.89 | 'user'=3.30 'nice'=0.50 'sys'=3.30 'softirq'=0.50 'irq'=0.50 'idle'=94.89
```

Of course this addon can handle warning and critical arguments (-w value -c value):

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_cpu.sh -w 5 -c 10
WARNING - user: 2.35, nice: 0.50, sys: 4.20, iowait: 0.50, irq: 0.50, softirq: 0.50 idle: 94.94 | 'user'=2.35 'nice'=0.50 'sys'=4.20 'softirq'=0.50 'irq'=0.50 'idle'=94.94
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_cpu.sh -w 2 -c 5
CRITICAL - user: 2.36, nice: 0.50, sys: 3.30, iowait: 0.50, irq: 0.50, softirq: 0.50 idle: 95.82 | 'user'=2.36 'nice'=0.50 'sys'=3.30 'softirq'=0.50 'irq'=0.50 'idle'=95.82
```

b. check_mem.pl

Type `./check_mem.pl -w <value> -c <value>`

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_mem.pl -w 50 -c 56
OK: Memory Usage (W> 50, C> 56): 50% <br>Swap Usage (W> 100, C> 100): 0%|MemUsed=50%;50;56 SwapUsed=0%;100;100
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_mem.pl -w 45 -c 56
<b>WARNING: Memory Usage (W> 45, C> 56): 50% <br>Swap Usage (W> 100, C> 100): 0%</b>|MemUsed=50%;45;56 SwapUsed=0%;100;100
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_mem.pl -w 35 -c 40
<b>CRITICAL: Memory Usage (W> 35, C> 40): 50% <br>Swap Usage (W> 100, C> 100): 0%</b>|MemUsed=50%;35;40 SwapUsed=0%;100;100
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins #
```

c. check_raid⁷

Type `./check_raid`. We'll get error ;)

„Can't locate IPC/Open2.pm in @INC (@INC contains: /usr/lib/perl5/site_perl/5.14.2/i486-linux /usr/lib/perl5/site_perl/5.14.2 /usr/lib/perl5/5.14.2/i486-linux /usr/lib/perl5/5.14.2 .) at ./check_raid line 372.

BEGIN failed--compilation aborted at ./check_raid line 372". We have to do two things:

1. create IPC directory at `/usr/lib/perl5/site_perl/<perl_version>/i486-linux/` and add it 755 mask.

```
mkdir /usr/lib/perl5/site_perl/<perl_version>/i486-linux/IPC
```

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # mkdir /usr/lib/perl5/site_perl/5.14.2/i486-linux/IPC
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # chmod 755 /usr/lib/perl5/site_perl/5.14.2/i486-linux/IPC
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ls -Al /usr/lib/perl5/site_perl/5.14.2/i486-linux/IPC
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ls -Al /usr/lib/perl5/site_perl/5.14.2/i486-linux/
total 424
drwxr-xr-x 12 root root 4096 Feb 25 07:07 auto
drwxr-xr-x  3 root root 4096 Feb 25 07:04 Compress
drwxr-xr-x  2 root root 4096 Feb 13 14:44 DBD
-r--r--r--  1 root root 313695 Oct 28 13:45 DBI.pm
drwxr-xr-x  2 root root 4096 Feb 13 14:44 Digest
drwxr-xr-x  2 root root 4096 Feb 13 14:44 GD
-r--r--r--  1 root root 60666 Jul  2 2013 GD.pm
drwxr-xr-x  2 root root 4096 Feb 13 14:44 HTML
drwxr-xr-x  2 root root 4096 Mar 31 20:11 IPC
```

2. Copy two files to IPC directory: Open2.pm and Open3.pm⁸

```
cp ~/Open2.pm /usr/lib/perl5/site_perl/5.14.2/i486-linux/IPC
```

```
cp ~/Open3.pm /usr/lib/perl5/site_perl/5.14.2/i486-linux/IPC
```

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # cp ~/Open* /usr/lib/perl5/site_perl/5.14.2/i486-linux/IPC/
```

Now when we type `./check_raid` we'll get informations about software RAID⁹¹⁰.

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_raid
OK: mdstat:[md1(55.18 GiB raid1)::UU, md0(766.94 MiB raid1)::UU]
```

d. check_disk

Type `./check_disk -w <value> -c <value>`

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_disk -w 40 -c 40
DISK OK - free space: /dev 501 MB (99% inode=99%); / 46 MB (6% inode=72%); /var/log 52432 MB (99% inode=99%); /tmp 501 MB (100% inode=99%); /dev/shm 501 MB (100% inode=99%); /dev=0MB;471;461;0;501 /dev=670MB;724;714;0;754 /var/log=360MB;55579;55579;0;55619 /tmp=0MB;471;461;0;501 /dev/shm=0MB;471;461;0;501
```

You'll get informations about all disk, if you want get informations about specific disk you'll have to type:

`./check_disk -w <value> -c <value> -p path (or -x device)`

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_disk -w 40 -c 41 -p /dev/md0
DISK OK - free space: / 46 MB (6% inode=72%); /dev=670MB;714;713;0;754
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ./check_disk -w 40 -c 41 -p /dev/md1
DISK OK - free space: /var/log 52432 MB (99% inode=99%); /var/log=360MB;55579;55578;0;55619
```

⁷ When RAID is OK then it returns OK (for all RAIDs). When one of disk is missing and RAID is broken it returns CRITICAL. When RAID is rebuilding it returns WARNING (with % of rebuild RAID).

⁸ When I added Open2.pm file and run that command I get error of Open3.pm missing file.

⁹ At IPCop Router we can configure software RAID at installation time. I do not know if IPCop works with hardware RAID.

¹⁰ Nagios can handle software RAID checking. This plugin can get informations about hardware RAID but Nagios returns UNKNOWN: dmraid[Plugin error]. At the end of this file I added screens from test alerts send via e-mail.

Now we need add some commands definitions at *nrpe.cfg* file. Open it and add what you want.

```
vi /var/ipcop/addons/nrpe/nrpe.cfg
```

Add some commands

```
command[check_sw_raid]=/var/ipcop/addons/nrpe/plugins/check_raid
```

```
command[check_/]=/var/ipcop/addons/nrpe/plugins/check_disk -w 20% -c 10% -p /dev/md0
```

```
command[check_/var/log]=/var/ipcop/addons/nrpe/plugins/check_disk -w 20% -c 10% -p /dev/md1
```

```
command[check_mem]=/var/ipcop/addons/nrpe/plugins/check_mem.pl -w 98 -c 99
```

```
command[check_cpu]=/var/ipcop/addons/nrpe/plugins/check_cpu.sh -w 50 -c 85
```

Save and close *nrpe.cfg* file (:wq [Return]).

D. Run NRPE daemon

After installation in */etc/rc.d/* directory was created file *rc.nrpe*. If we want run NRPE daemon we need type */etc/rc.d/rc.nrpe start* (stop|restart). Now we need check if is it working, type *ps aux |grep nrpe* we should see information that NRPE is working.

And we need to know if IPCop is listening at 5666 port – *netstat -an |grep 5666*.

```
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # ps aux |grep nrpe
nagios 6517 0.0 0.0 3948 976 ?        Ss   20:39   0:00 /var/ipcop/addons/nrpe/nrpe -c /var/ipcop/addons/nrpe/nrpe.cfg -d
root    6753 0.0 0.0 5704 516 pts/0    S+   20:41   0:00 grep nrpe
root@router-ld-toya:/var/ipcop/addons/nrpe/plugins # netstat -an |grep 5666
tcp      0      0 192.168.5.2:5666    0.0.0.0:*        LISTEN
```

2. Nagios Server

A. Test if everything is working.

a. `check_nrpe`

When we're logged in Nagios Server via ssh we can check if Nagios Server is capable to connect to monitored IPCop Router. All we have to do is type `/usr/local/nagios/libexec/check_nrpe -H <IP of IPCOP>` and we should get „NRPE v2.15”.

```
tai@nagios:~$ /usr/local/nagios/libexec/check_nrpe -H 1'2.1' .4.3
NRPE v2.15
```

b. `check_nrpe` with `check_command`

We can check if our commands working correctly before we start adding IPCop host to our Nagios configuration file. Type

```
/usr/local/nagios/libexec/check_nrpe -H <IP of IPCOP> -c <check_command>.
```

It's important that `<check_command>` is the same as `command[check_command]` in `nrpe.cfg` file at IPCop Router!

For example, if we declader (at IPCop Router in `nrpe.cfg` file):

Command[**check_sw_raid**]=/path/to/check/command/addon/check_raid
we need to type at Nagios:

```
/usr/local/nagios/libexec/check_nrpe -H <IP of IPCOP> -c check_sw_raid.
```

```
tai@nagios:~$ /usr/local/nagios/libexec/check_nrpe -H 1'2.1' .4.3 -c check_sw_raid
OK: mdstat:[md1(55.27 GiB raid1):UU, md0(638.94 MiB raid1):UU]
```

B. Host/Service configuration.

Now we need add our IPCop Router to be monitored by Nagios Server. Note that `linux-box` is template created by me. You may be willing using default linux template.

```
# tmp
define host{
    use          generic-host
    name         linux-box
    register     0
    check_period 24x7
    max_check_attempts 1
    normal_check_interval 5
    retry_check_interval 1
    contact_groups admins
    check_command check-host-alive
    notification_interval 30
    notification_period 24x7
}
```


a. Host configuration.

Open file where you want store your IPCop Router declarations. This is example of my host declaration:

```
define host{
    use                linux-box
    host_name          router-wa-vectra
    alias              Lacze Vectra TAIWA
    address            172.17.24.3
    icon_image         ipcop.png
    statusmap_image    ipcop.png
    parents            Srv Switch
}
```

b. Service configuration.

At the same file add declaration for service(s) to be monitored by Nagios Server. This is example of my services declarations:

Monitoring Software RAID at Linux Servers

```
define service{
    use                local-service
    host_name          localhost, router-wa-vectra
    service_description SW RAID Status
    check_command      check_nrpe!check_sw_raid
    notifications_enabled 1
    flap_detection_enabled 0
    active_checks_enabled 1
    passive_checks_enabled 0
}
```

Monitoring HDD partitions

/

```
define service{
    use                local-service
    host_name          domena-wa, localhost, router-wa-vectra
    service_description /
    check_command      check_nrpe!check_/
    notifications_enabled 1
    flap_detection_enabled 0
    active_checks_enabled 1
    passive_checks_enabled 0
}
```

/var/log - IPCop

```
define service{
    use                local-service
    host_name          router-wa-vectra, ipcop-test
    service_description /var/log
    check_command       check_nrpe!check_/var/log
    notifications_enabled 1
    flap_detection_enabled 0
    active_checks_enabled 1
    passive_checks_enabled 0
}
```

Memory Usage

```
define service{
    use                local-service
    host_name          localhost, domena-wa, router-wa-vectra
    service_description MEM Usage
    check_command       check_nrpe!check_mem
    notifications_enabled 1
    flap_detection_enabled 0
    active_checks_enabled 1
    passive_checks_enabled 0
}
```

CPU Usage


















```
define service{
    use                local-service
    host_name          localhost, domena-wa, router-wa-vectra
    service_description CPU Usage
    check_command       check_nrpe!check_cpu
    notifications_enabled 1
    flap_detection_enabled 0
    active_checks_enabled 1
    passive_checks_enabled 0
}
```

After we do everything, we need to check Nagios configuration and restart service.

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
service nagios restart
```

Now we should be able to see IPCop and monitored services in our Nagios WWW Console.

Serwery Linuksowe TAIWA (linux-servers-wa)

Host	Status	Services	Actions
	UP	7 OK 1 WARNING	  
ipcop-test	DOWN	5 OK 1 CRITICAL	  
localhost	UP	5 OK	  
	UP	1 OK	  
router-wa-vectra	UP	6 OK	  

Host	Service	Status	Last Check	Duration	Attempt	Status Information
router-wa-vectra	/	OK	31-03-2014 21:09:00	0d 22h 0m 16s	1/4	DISK OK - free space: / 145 MB (24% inode=70%);
	/varlog	OK	31-03-2014 21:09:03	0d 22h 0m 13s	1/4	DISK OK - free space: /varlog 52857 MB (99% inode=99%);
	CPU Usage	OK	31-03-2014 20:51:59	0d 9h 17m 16s	1/4	OK - user: 0.50, nice: 0.50, sys: 6.92, lowat: 0.50, irq: 0.50, softirq: 0.50 idle: 94.07
	MEM Usage	OK	31-03-2014 21:09:08	0d 22h 0m 6s	1/4	OK: Memory Usage (W> 98, C> 99): 27% Swap Usage (W> 100, C> 100): 0%
	PING	OK	31-03-2014 20:50:43	11d 12h 9m 59s	1/4	PING OK - Packet loss = 0%, RTA = 0.14 ms
	SW RAID Status	OK	31-03-2014 20:49:15	0d 22h 0m 1s	1/4	OK: mdstat[md1(55.27 GiB raid1):UU, md0(638.94 MB raid1):UU]

C. RAID notifications.

RAID CRITICAL

**** PROBLEM Usługa: kontroler domeny TAIWN/SW RAID Status ma stan CRITICAL ****

nagios@nagios

Wysłano: Cz 2014-03-27 12:46

Do: ~~test@tai.pl~~ ~~test@tai.pl~~

***** Nagios *****

Typ powiadomienia: PROBLEM

Usługa: SW RAID Status

Host: kontroler domeny TAIWN

IP: ~~192.168.1.1~~

Stan: CRITICAL

Data/Godzina: Thu Mar 27 12:46:06 CET 2014

Informacje dodatkowe:

CRITICAL: mdstat:[md2(279.40 GiB raid1):UU, md3(138.40 GiB raid1):F:sdb7:U_, md0(46.57 GiB raid1):UU, md1(1.40 GiB raid1):UU, md127(931.52 GiB raid0):OK]

RAID WARNING

**** PROBLEM Usługa: kontroler domeny TAIPN/SW RAID Status ma stan WARNING ****

nagios@nagios

Wysłano: Cz 2014-03-27 12:48

Do: alert@taipn.pl, mihod@taipn.pl

***** Nagios *****

Typ powiadomienia: PROBLEM

Usługa: SW RAID Status

Host: kontroler domeny TAIPN

IP: 193.168.1.4

Stan: WARNING

Data/Godzina: Thu Mar 27 12:48:11 CET 2014

Informacje dodatkowe:

WARNING: mdstat:[md2(279.40 GiB raid1):UU, md3(138.40 GiB raid1):U_ (recovery:1.0% 57137K/sec
ETA: 41.8min), md0(46.57 GiB raid1):UU, md1(1.40 GiB raid1):UU, md127(931.52 GiB raid0):OK]

RAID OK

**** RECOVERY Usługa: kontroler domeny TAIPN/SW RAID Status ma stan OK ****

nagios@nagios

Wysłano: Cz 2014-03-27 13:41

Do: alert@taipn.pl, mihod@taipn.pl

***** Nagios *****

Typ powiadomienia: RECOVERY

Usługa: SW RAID Status

Host: kontroler domeny TAIPN

IP: 193.168.1.4

Stan: OK

Data/Godzina: Thu Mar 27 13:40:50 CET 2014

Informacje dodatkowe:

OK: mdstat:[md2(279.40 GiB raid1):UU, md3(138.40 GiB raid1):UU, md0(46.57 GiB raid1):UU, md1(1.40
GiB raid1):UU, md127(931.52 GiB raid0):OK]