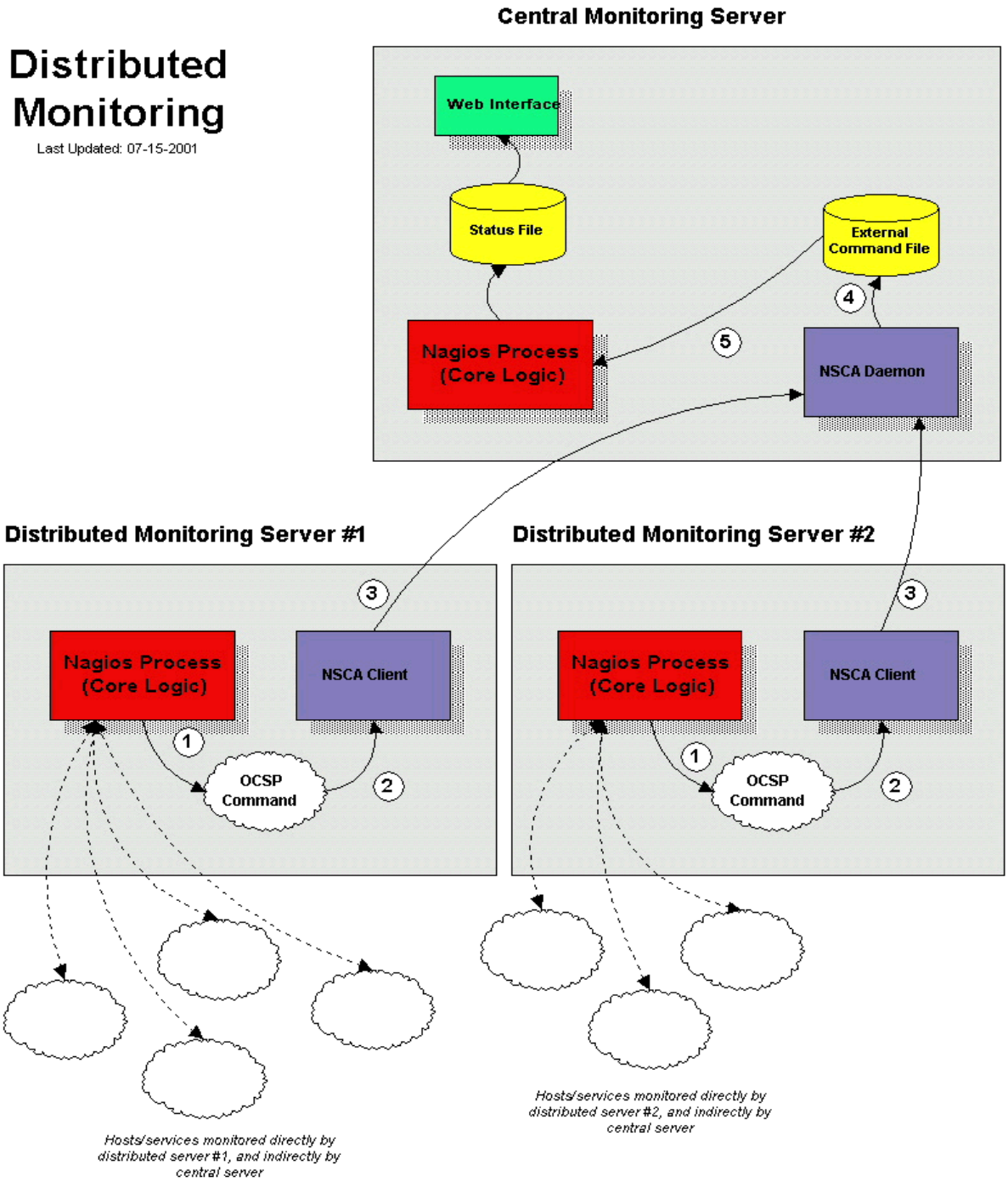
Nagios NSCA Indirect Monitoring, Passive Check



NSCA passzív monitoring

Az **NSCA**-val végrehajtott passive check monitoringnak a lényege az ábrán jól látszódik. A központi Nagios nem küld (aktív) check parancsokat, hanem az NSCA-n keresztül passzívan várja a SLAVE (kliens) Nagios-tól a check eredményeket. A Központi Nagios (MASTER, server) csak megjeleníti az eredményt. Az **NSCA** abban különbözik az *NRPE* rendszerhez képest, hogy az NSCA esetén egy távoli, kívülről nem elérhető(!) hálózatot is tudunk monitorozni. Ebben az esetben az NSCA oldalán futó hálózatban kell megnyitni az 5667 tcp portot a bejövő adatokhoz!
Lehetőség szerint az *nsca* verziók egyezzenek a master (central), slave (client) szervereken!

NSCA szerver (MASTER)

- A Master nagios-ra az nsca csomag telepítése szükséges (ill. az xinetd) alapbeállításokkal:

yum install nsca xinetd

- Az **nsca** xinetd-ből indul, ezt beállíthatjuk az ***/etc/xinetd.d/nsca*** fájllal:

```
# default: on
# description: NSCA
service nsca
{
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = nagios
        group           = nagios
        server          = /usr/sbin/nsca
        server_args     = -c /etc/nagios/nsca.cfg --inetd
        log_on_failure  += USERID
        disable         = no
        #only_from      = <ipaddress1> <ipaddress2>
}
```

- A szolgáltatást indítsuk is el:

service xinetd start

- Az */etc/services* állományban beállíthatjuk a „***nsca 5667/tcp # NSCA Nagios***” sort.
- A monitorozott hosztot definiálni kell mindkét Nagios-on:

```
define host {
   use                  linux-server;
   host_name            www.mysite.hu;
   address              www.mysite.hu;
}
# Generic service definition template
define service{
   name                         generic-service ;
   active_checks_enabled        0       ;
   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       ;
   process_perf_data            1       ;
   register                     0       ;
}

# Service definition for http
define service{
    use                         generic-service;
    host_name                   www.mysite.hu;
    service_description         HTTP    ;
    is_volatile                 0       ;
    check_period                24x7    ;
    max_check_attempts          3       ;
    normal_check_interval       1       ;
    retry_check_interval        5       ;
    #contact_groups             admins,webmaster        ;
```

```
    notification_options            w,u,c,r   ;
    notification_interval           960       ;
    notification_period             24x7      ;
    check_command                   check_http        ;
}


# /etc/nagios/nagios.cfg
accept_passive_service_checks=1
passive_checks_enabled=1
```

NSCA Kliens (SLAVE)

- A Kliensen (SLAVE) az nsca-client csomag kell (alapbeállítás maradhat):

yum install nsca-client

- A kliens Nagios-t is konfigoljuk:

```
# Generic service definition template
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           0       ;
    event_handler_enabled           1       ;
    flap_detection_enabled          1       ;
    process_perf_data               1       ;
    register                        0       ;
}

# Service definition for http
define service{
        use                             generic-service;
        host_name                       www.mysite.hu;
        service_description             HTTP;
        is_volatile                     0;
        check_period                    24x7;
        max_check_attempts              3;
        normal_check_interval           1;
        retry_check_interval            5;
        # contact_groups                admins,webmaster;
        notification_options            w,u,c,r;
        notification_interval           960;
        # notification_period           never;
        notification_period             24x7;
        check_command                   check_http;
}

define host {
    use                     linux-server;
    host_name               www.mysite.hu;
    address                 www.mysite.hu;
}
```

```
# /etc/nagios/nagios.cfg
 obsess_over_services=1
 ocsp_command=submit_service_check
 ocsp_timeout=5
 obsess_over_hosts=1
 ochp_command=submit_host_check
 ochp_timeout=5

# /etc/nagios/conf.d/commands.cfg
define command {
    command_name    submit_service_check
    #command_line    /usr/lib64/nagios/plugins/submit_service_check.sh $HOSTNAME$
'$SERVICEDESC$' $SERVICESTATEID$ '$SERVICEOUTPUT$'
    command_line    $USER1$/submit_service_check.sh $HOSTNAME$ '$SERVICEDESC$'
$SERVICESTATEID$ '$SERVICEOUTPUT$'
}


define command {
    command_name    submit_host_check
    command_line     $USER1$/submit_host_check.sh $HOSTNAME$ $HOSTSTATEID$
'$HOSTOUTPUT$'
}
```

- A ***submit_service_check.sh*** szkript az alábbi:

```
#!/bin/bash
# /usr/bin/printf "%s\t%s\t%s\t%s\n" "$1" "$2" "$3" "$4" | /usr/lib64/nagios/plugins/send_nsca -H
<IP_or_host_name> -c /etc/nagios/send_nsca.cfg
```

Pl.:

```
#!/bin/bash
# /usr/lib64/nagios/plugins/submit_service_check.sh
/usr/bin/printf "%s\t%s\t%s\t%s\n" "$1" "$2" "$3" "$4" | /usr/sbin/send_nsca -H 10.10.10.116 -c
/etc/nagios/send_nsca.cfg
```

A fenti szkript elküldi a **send_nsca** segítségével a központi Nagios-nak a check adatokat (minden check lefutása után automatikusan). Fontos a szkriptben levő \n jel a sorvégére, mert soronként értelmezi az adatokat az nsca.

- A ***submit_host_check.sh*** az alábbi lehet pl.:

```
#!/bin/bash
# Arguments:
#  $1 = host_name (Short name of host that the service is
#       associated with)
#  $2 = state_string (A string representing the status of
#       the given service - "UP", "DOWN", "UNREACHABLE"
#  $3 = plugin_output (A text string that should be used
#       as the plugin output for the service checks)
#
# pipe the service check info into the send_nsca program, which
# in turn transmits the data to the nsca daemon on the central
# monitoring server
#!/bin/sh
return_code=-1
case "$2" in
            UP)
                 return_code=0
                    ;;
            DOWN)
                 return_code=1
                        ;;
            UNREACHABLE)
                 return_code=2
                        ;;
        esac

# Test everything works fine
/usr/bin/printf "%s\t%s\t%s\n" "$1" "$return_code" "$3" |
/usr/sbin/send_nsca -H <IP_or_host> -p PORT -c
/etc/nagios/send_nsca.cfg
```

„Check freshness” opció a MASTER szerveren

Egy fontos dolgot még célszerű beállítani a Nagios Szerveren (MASTER). Ha az *nsca kliens* nem elérhető (megszakad a kapcsolat, vagy meghibásodik), akkor nem küld check adatokat a MASTER felé. Ezt a MASTER szerver nem fogja észrevenni, ill. azt látszódik, hogy hosszú idő óta nem változott a beállított szolgáltatás(ok) állapota (a legutóbbi marad meg aktuálisnak). Ezért a „check freshness” opciót kell bekapcsolnunk a szolgáltatások definíciójában, pl.:

```
define service{
        host_name               server
        service_description     ArcServe Job
        active_checks_enabled   0       ; active checks are NOT enabled
        passive_checks_enabled  1       ; passive checks are enabled (this is how results are reported)
        check_freshness         1
        freshness_threshold     93600     ; 26 hour threshold, since backups may not always finish at the same time
        check_command           no-report ; this command is run only if the service results are "stale"
        ;...other options...
        }

define command{
        command_name    no-report
        command_line    /usr/local/nagios/libexec/check_dummy 2 "CRITICAL: No check report!"
        }
```

Tehát definiálnunk kell egy „dummy” parancsot (pl. ***no-report),*** ami a megadott státusz és szöveggel tér vissza. A ***service*** részben definiálni kell a „check_fressness” opciót (értéke: 1), a „freshness_threshold” opciót (értéke: <*másodpercek*>).

A fenti példában tehát 26 óránként (93600mp) figyeli a Master Nagios, hogy az állapot ellenőrzés megküldése megtörtént-e. Ha nem, akkor a ***no-report*** parancs lefut és visszaadja a megadott állapotot és szöveget, pl.: ***CRITICAL: No check report!***