v1.16_keepalived_1.3.6

keepalived 1.3.6

, 1.2.15 , 1.3.6

keepalived open source , (high availabilitty) (load-balancing). VRRP, Linux Vitrual Server (IPVS). Keepalived Eltex , , GitHub (https://github.com/acassen/keepalived) .

Keepalived SoftWLC, VRRP.

keepalived Ubuntu 14.04 Wi-Fi .

:

```
root@master:/# dpkg -i keepalived_1.3.6-eltexu14_amd64.deb
```

root

/

:

```
service keepalived start
```

:

```
keepalived start/running, process 2471
```

:

```
root@master:/# service keepalived stop
```

:

```
keepalived stop/waiting
```

:

```
root@master:/# service keepalived status
```

:

```
keepalived start/running, process 2809
```

keepalived

/etc/keepalived/keepalived.conf	
/etc/keepalived/check_ping.sh	EMS
/etc/keepalived/keep_notify.sh	, (MASTER, BACKUP, FAULT)
/etc/sysconfig/keepalived	keepalived
/etc/keepalived/mongo_switch.js	replicaSet MongoDB , VRRP

/etc/keepalived/keepalived.conf

```
! Configuration File for keepalived

global_defs {
    notification_email {
       admin@example.org
    }
    notification_email_from softwlc@example.org
    smtp_server mail.example.org
    smtp_connect_timeout 30
    router_id swlc1
    enable_traps
}

vrrp_script check_network {
    script "/etc/keepalived/check_ping.sh"
    interval 5
    weight 50
    fall 3
    rise 3
    init_fail
    user root
}

vrrp_instance VI_SWLC {
    state BACKUP
    interface eth0
    virtual_router_id 1
    track_script {
        check_network
    }
    track_interface {
        eth0 weight 50
    }
    priority 150
    advert_int 1
    nopreempt
    authentication {
        auth_type PASS
        auth_pass eltex
    }
    virtual_ipaddress {
        <virtual_ip> dev eth0 label eth0:1
    }

    notify_master "/etc/keepalived/keep_notify.sh master"
    notify_backup "/etc/keepalived/keep_notify.sh backup"
    notify_fault "/etc/keepalived/keep_notify.sh fault"

    unicast_peer {
        <ip_server1>
    }
}
```

: **global_defs, vrrp_script, vrrp_instance.** , , VRRP .

-

:

/etc/keepalived/check_ping.sh

```
#!/bin/bash

# host to ping
# there - default gw
HOST=<default_gw_ip>
# -q quiet
# -c nb of pings to perform
ping -q -c5 $HOST > /dev/null

# $? var keeping result of execution
# previous command
if [ $? -eq 0 ]
    then
        echo `date +"%T %F"` "OK gw reachable"
        EXIT_CODE=0
    else
        echo `date +"%T %F"` "ERROR gw unreacheble!"
        EXIT_CODE=1
fi

exit $EXIT_CODE
```

. , , SoftWLC .

keep_notify.sh

/etc/keepalived/keep_notify.sh

```
#!/bin/bash

MYSQL_USER="<mysql_user>"
MYSQL_PASSWORD="<mysql_password>"

mongo_set_role() {
    local role="$1"
    if [[ "$(which mongo)" ]]; then
        mongo --quiet --eval "var role=\"$role\"" admin /etc/keepalived/mongo_switch.js
        # Uncomment if using mongodb auth
        #mongo -u<username> -p<password> --quiet --eval "var role=\"$role\"" admin /etc/keepalived/mongo_switch.
js
    fi
}

if ! lockfile-create --use-pid -r 5 /tmp/keep.mode.lock; then
    echo "Unable to lock"
    echo "Unable to lock" > /tmp/keep.mode.lock.fail
    exit 0
fi

case "$1" in
    master)
    #  ems_reload_all
    echo "MASTER" > /tmp/keep.mode

    mongo_set_role master
    service eltex-ems restart
    service tomcat7 restart
    service eltex-ngw restart

    #   MySQL     -   ,
    #     heartbeat
    mysql -u$MYSQL_USER -p$MYSQL_PASSWORD -e "stop slave"
    mysql -u$MYSQL_USER -p$MYSQL_PASSWORD -e "start slave"
  ;;
 backup)
    echo "BACKUP" > /tmp/keep.mode
    mongo_set_role slave
    service mongodb restart
    service eltex-ems stop
    service tomcat7 stop
    service eltex-ngw stop
    mysql -u$MYSQL_USER -p$MYSQL_PASSWORD -e "stop slave"
    mysql -u$MYSQL_USER -p$MYSQL_PASSWORD -e "start slave"
 ;;
 fault)
    echo "FAULT" > /tmp/keep.mode
    mongo_set_role slave
    service mongodb restart
 ;;
 *)
    echo "Usage: $0 {master|backup|fault}"
    exit 1
esac

lockfile-remove /tmp/keep.mode.lock;

exit 0
```

replicaSet MongoDB.

/etc/keepalived/mongo_switch.js

```
//
var role;

if (role != 'master' && role != 'slave') {
    throw "Role must be either master or slave";
}

var thisIsMaster = (role == 'master');
var status = rs.isMaster();
var thisHost = status.me;

print("Primary: " + status.ismaster + "; applying configuration ...");
var cfg = rs.conf();
for (var i = 0; i < cfg.members.length; i++) {
    var member = cfg.members[i];
    var self = (member.host == thisHost);
    if (self ^ thisIsMaster) {
        //   slave
        member.priority = 1;
        member.votes = 0;

        print(member.host + ": secondary");
    } else {
        //   master
        member.priority = 2;
        member.votes = 1;

        print(member.host + ": primary");
    }
}

var result = rs.reconfig(cfg, { force: !status.ismaster });
if (result.ok == 1) {
    print("Reconfiguration done");
} else {
    print(result);
}
```

keepalived

etc

keepalived **/var/log/syslog** . , keepalived , -. **rsyslog**:

```
nano -w /etc/rsyslog.d/10-keepalived.conf
if $programname contains 'Keepalived' then /var/log/keepalived.log
if $programname contains 'Keepalived' then ~
```

rsyslog :

```
root@swlc01-server:/# service rsyslog restart
```

keepalived - **/var/log/keepalived.log** **/var/log/syslog** .