

Wysoka dostępność dla sieci VPN

INNymi DROGAMI

Protokół IPSec uniemożliwia wykorzystanie wielu sprytnych sztuczek, stosowanych przez produkty cechujące się wysoką dostępnością. Pokażemy rozwiązanie, które zapewni przezroczyste łącze zapasowe dla połączeń IPSec.

JOHANNES HUBERTZ

Administratorzy systemu często wymagają systemu połączeń sieciowych, który przełącza się w sposób przezroczysty na połączenie zapasowe, kiedy połączenie podstawowe ulegnie awarii. Jeśli jednak przekazywane przez Internet dane chronimy za pomocą wirtualnej sieci prywatnej w protokole IPSec, wówczas łącza zapasowemu musimy poświęcić szczególną uwagę.

Jest tak dlatego, że IPSec [1] [2] wymaga stałych adresów IP na obu końcach tunelu. Kiedy więc sieć przełączy się na inny tunel, adresy IP muszą zmienić się zgodnie z nowymi punktami końcowymi, gdyż w przeciwnym razie istniejące połączenia zostaną zakończone. Protokół BGP (Border Gateway Protocol [3]) zapewnia niezawodną metodę utrzymania wysokodostępnej puli adresów IP u wielu usługodawców. Niestety, umowy z usługodawcami często uniemożliwiają administratorom użycie protokołu BGP dla istniejącego połączenia internetowego.

Aby ominąć ten problem, wielu administratorów obchodzi się bez automatu, a gdy nastąpi najgorsze, ręcznie przełączają oni łącze ze standardowego na zapasowe. Często wymaga to nawet fizycznego przekładania kabli między interfejsami. Nie jest to najnowocześniejsze rozwiązanie na miarę naszych czasów. Byłoby lepiej, gdyby urządzenia sieciowe wykrywały awarię łącza i przełączały się automatycznie. Idealny system automatycznie generowałby także ustawienia dla obu końców połączenia, odwołując się do centralnej konfiguracji.

Dla zapór i bram IPSec konfiguracja centralna stanowi ultranowoczesną technologię. W Linuksie obsługuje to SSPE (Simple Security Policy Editor [4]). Jednak wysokodostępne rozwiązanie opisane w tym artykule nie obsługuje jeszcze SSPE.

Linux-HA

Projekt Linux-HA [5] jest przeznaczony dla wysokodostępnych rozwiązań serwerów liniuksowych. To oprogramowanie umożliwia administratorom skonfigurowanie wysokodostępnej sieci VPN (która nie wymaga protokołu BGP), która przełączy się szybciej z trybu standardowego na zapasowy. Aby wdrożyć to rozwiązanie, potrzebujemy dwóch niezależnych, równoległych tuneli; każdorazowo będziemy korzystać tylko z jednego tunelu. Każdy tunel w każdej sieci ma swój punkt końcowy, stanowiący bramę do sieci lokalnej. W oprogramowaniu Linux-HA zaimplementowano automatyczną rekonfigurację adresów IP w celu obsługi takiej konfiguracji.

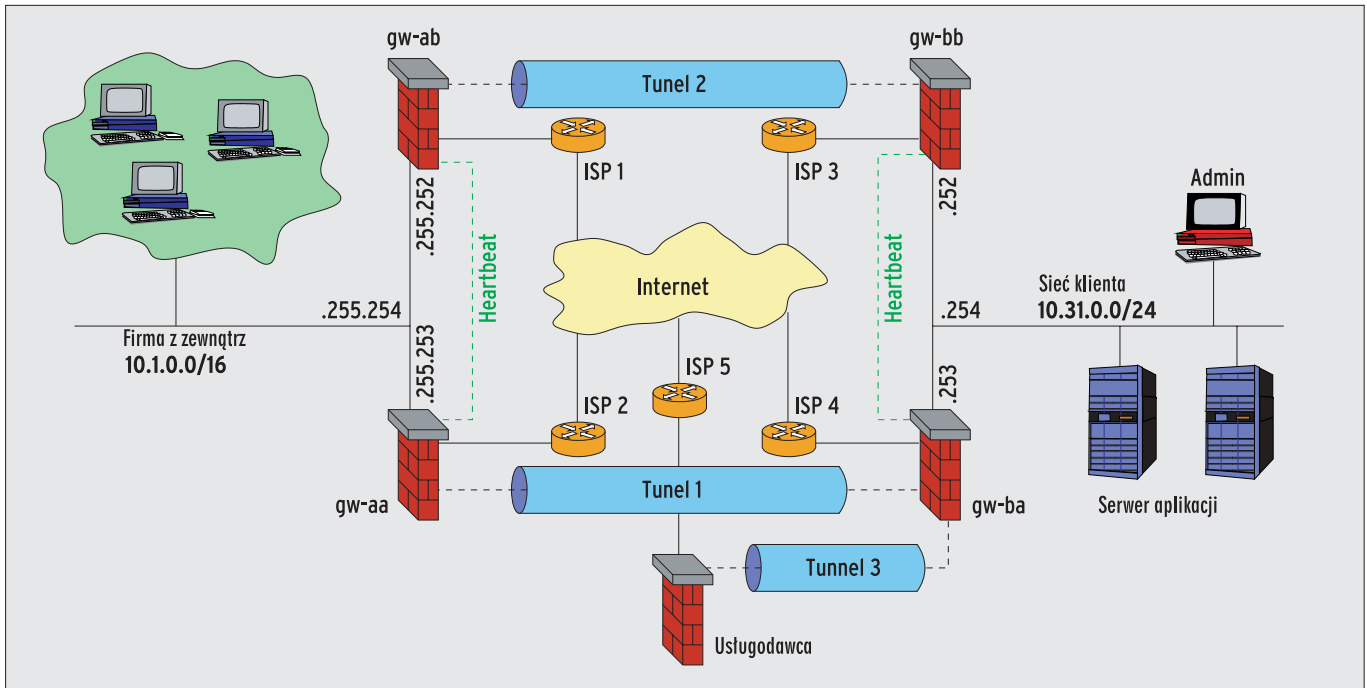
Dwa węzły wysokodostępne mają zarówno adres indywidualny, jak i wspólny. Adres wspólny jest używany każdorazowo tylko przez jeden komputer. Mechanizm taki jest przeznaczony dla serwerów (WWW, pocztowych) ze wspólnym adresem. Usługa działa na obu maszynach i oczekuje na każdym adresie IP; żądania napływają jednak tylko przez wspólny adres IP. Dzięki temu oprogramowanie Linux-HA przydziela w razie awarii dru-

giemu komputerowi adres zewnętrzny, a w idealnej sytuacji użytkownicy nawet nie zauważą nieaktywności pierwszego.

Kabel szeregowy zapewnia sygnał testujący dla obu komputerów. Sygnał testujący stanowi istotną część oprogramowania Linux-HA, gdyż komputery za jego pomocą sprawdzają dostępność swoich węzłów równorzędnych. Jeśli jeden komputer ulegnie awarii, komputer nadrzędny przyjmuje adres IP należący do pierwszego komputera. Generuje on emisje ARP za pomocą wspólnego adresu IP i własnego adresu MAC w sieci lokalnej (a więc realizuje on coś w rodzaju legalnego fałszowania pakietów ARP). Co więcej, komputer umożliwia stosowanie aliasu interfejsu. Kiedy tylko pierwszy węzeł będzie znowu w trybie on-line, protokół sygnału testującego zapewni, że serwer 2 znowu wyłączy interfejs aliasowy, podczas gdy serwer 1 włączy swój interfejs i emisje ARP w sieci lokalnej.

Dwie instalacje Linux-HA

W układzie widocznym na Rysunku 1 klient znajduje się z lewej strony, zaś firma zewnętrzna z prawej. Strona klienta (po lewej stronie, *gw-aa*) i strona usługodawcy bramy IPSec (po prawej, *gw-ba*) przesyłają pakiety w tunelu za pomocą protokołu ESP (Encapsulating Security Payload, protokół z grupy IPsec). Tunel zastępczy (2) jest skonfigurowany na dwóch zapasowych ruterach na górze (*gw-ab*, *gw-bb*) w sposób podobny do tunelu 1.



Rysunek 1: W tej sytuacji sieć klienta jest podłączona do usługodawcy z zewnątrz poprzez sieć VPN. To połączenie korzysta z dwóch alternatywnych ścieżek; w razie konieczności tunel 2 przejmuje rolę tunelu 1. Inny usługodawca jest podłączony, lecz nie dysponuje funkcjami HA.

Listing 1: Nadzorca HA-VPN

```
#!/bin/bash
# Nadzorca HA-VPN w węźle gw-aa
# Drugi koniec tunelu
TARGET="gw-ba"
# Liczba sekund między sygnałami ping
TIMEOUT=1
# Oczekiwanie przez MAXFAIL *
TIMEOUT przed włączeniem cofnięcia
MAXFAIL=5
# Oczekiwanie HYSTERE * TIMEOUT
po zakończeniu awarii
# zanim skrypt powróci do
normalnego działania
HYSTERE=180
# Założenie: brak błędów przy
uruchomieniu
FAIL=0
VERBOSE=""
ACTION_FAIL_START="/root/bin/
HA-VPN-action-script start"
ACTION_OK_AGAIN="/root/bin/
HA-VPN-action-script stop"
PING=/usr/bin/echo ping
LOG="/usr/bin/logger -t HA-VPN"
math () {
    eval echo "\${($*)}"
}
echo "`date +%Y%m%d%H%M%S`
`basename $0` starting" | $LOG
while :
do
    VAL=`$PING ${VERBOSE} -u -t
    $TIMEOUT -s 5 ${TARGET} 2>&1`
    ERROR=$?
    if [ $ERROR -gt 0 ] ; then
        echo "$DAT $ERROR $FAIL
        $VAL" | $LOG
        # Przekroczenie limitu czasu
        if [ $FAIL -lt 0 ] ; then
            # Inny błąd w fazie
            powrotu do stanu normalnego
            FAIL=`math $MAXFAIL + 1`
        fi
        if [ $FAIL -eq $MAXFAIL ] ;
        then
            # Rozpoczęcie cofnięcia
            :
            FAIL=`math $FAIL + 1`
            echo "$DAT starting backup
            now: ${ACTION_FAIL_START}" | $LOG
            ${ACTION_FAIL_START}
        else
            if [ $FAIL -lt $MAXFAIL ] ;
            then
                FAIL=`math $FAIL + 1`
            fi
        fi
    else
        # Udany sygnał ping
        if [ $FAIL -gt $MAXFAIL ] ;
        then
            FAIL=`math 0 - $HYSTERE`
            fi
            if [ $FAIL -le $MAXFAIL
            -a $FAIL -ge 0 ] ; then
                FAIL=0
            fi
            if [ $FAIL -lt 0 ] ; then
                # Oczekiwanie przez okres
                histerezy przed ponownym
                uruchomieniem
                echo "$DAT $ERROR $FAIL
                $VAL" | $LOG
                FAIL=`math $FAIL + 1`
                if [ $FAIL -eq 0 ] ; then
                    # Przywrócenie
                    normalnego działania
                    :
                    echo "$DAT normal again
                    now: ${ACTION_OK_AGAIN}" | $LOG
                    ${ACTION_OK_AGAIN}
                fi
            fi
        fi
        #echo "$DAT $ERROR $FAIL $VAL" |
        $LOG
        sleep $TIMEOUT
        done
        # nigdy nie zdarza się:
        exit 0
    fi
done
```

Linux-HA działa między węzłami *gw-aa* i *gw-ab* oraz między *gw-ba* i *gw-bb*. Sygnał testujący wykorzystuje port szeregowy, który nie jest potrzebny do żadnego innego celu, a zatem jest niezależny od sieci, IPsec i IPtables. Obydwie instalacje HA pracują osobno i niezależnie od siebie. W normalnym trybie *gw-aa* i *gw-ba* mają adresy IP lokalnego rutera (10.1.255.254 z lewej strony i 10.31.0.254 z prawej). W wypadku awarii te adresy przechodzą do węzłów *gw-ab* i *gw-bb*. Aby umożliwić administratorom wyraźne podłączenie do komputerów w puli HA, obydwa mają dodatkowo adresy statyczne w ich odpowiednich sieciach lokalnych; np. 10.31.0.252 dla *gw-bb*.

Skrypt tworzący sieć HA-VPN

Domyślne bramy *gw-aa* i *gw-ba* uruchamiają skrypt powłoki, taki jak na Listingu 1. Skrypt jest uruchamiany przez wpis w pliku *inittab* i monitoruje dostępność drugiej strony, niezależnie od tunelu IPsec. W tym celu skrypt wysyła krótkie pakiety UDP do portu echo. Program echoping to standardowy składnik większości dystrybucji Linuksa; można użyć protokołu UDP, ustawiając odpowiednie opcje. Taka metoda pozwala uniknąć problemów wynikających ze zbyt żarłocznych zapór, które usuwają wszystkie komunikaty ICMP, a zatem blokują normalne sygnały ping.

Dopóki połączenie między węzłami *gw-aa* i *gw-ba* jest aktywne, licznik *FAIL* zawsze wynosi zero. Jeśli na sygnał ping nie ma odpowiedzi, skrypt zwiększa zmienną *FAIL* w wierszu

53, dopóki wartość jest niższa od stałej *MAXFAIL*. Jeśli następny sygnał ping dotrze, skrypt znowu ustawia licznik *FAIL* na zero (wiersz 62). Jeśli zmienna osiągnie wartość *MAXFAIL*, w wierszu 50 zostanie uruchomiony program *ACTION_FAIL_START* (Listing 2 z parametrem *start*). Program *ACTION_FAIL_START* wyłącza lokalny sygnał testujący, co powoduje automatycznie przyjęcie przez zapasową bramę adresu IP rutera lokalnego.

Oczekiwanie na normalne działanie

Nieskończona pętla działa i czeka na ponowną aktywację łącza, oczekując, że program echoping zwróci normalną wartość 0. Kiedy pierwsza odpowiedź dotrze po dłuższej awarii, wówczas łącze może być niecałkowicie stabilne, więc skrypt odczeka chwilę przed przywróceniem normalnych operacji na bramie. Ustawia zmienną *FAIL* na wartość ujemną *HYSTERE* (wiersz 59) i zwiększa *FAIL* dla każdego otrzymanego sygnału ping (wiersz 67). Jeśli inny błąd wystąpi w fazie powrotu do normalnego działania, w wierszu 43 zmienna *FAIL* zostanie ustawiona na wartość większą niż *MAXFAIL* – dzięki temu system będzie nadal używać łącza zapasowego.

Działanie nie powróci do stanu normalnego, dopóki licznik *FAIL* nie osiągnie ponownie zera. Wówczas skrypt w wierszu 71 wywoła program *ACTION_OK_AGAIN* (Listing 2 z parametrem *stop*). Ten program włączy także ponownie sygnał testujący, a zatem przywróci również adres IP lokalnego rutera.

Takie wielopoziomowe podejście umożliwia uniknięcie szybkiego przerzucania tras, kiedy bramy oscylują szybko między tunelami. Połączenia internetowe cechują się krótkimi przestojami działania, które zanikają po chwili. Pliki dzienników są pełne takich przykładów. Trzy minuty to wystarczająca wartość, aby zapewnić stabilność działania.

Doświadczenie w obsłudze tego rodzaju konfiguracji pokazuje, że awarie zawsze charakteryzowały się swoistą regularnością. Jeśli sygnał testujący na jednym końcu tunelu wyłączy bramę, drugi zwykle dołącza po co najwyżej sekundzie. Tego rodzaju analiza pliku dziennika zakłada, że na obu końcach połączenia zsynchronizowano zegary systemowe.

Satysfakcja gwarantowana

Opisane w tym artykule ustawienie jest aktywne od 2004 roku i wielokrotnie okazało się, że użytkownicy nie zauważają nawet awarii rutera po stronie usługodawcy. W wypadku całkowitej awarii normalne mechanizmy wykonywania ponownej próby, używane przez stosy protokołów TCP/IP na serwerach aplikacji i klienckich komputerach PC, mogą z łatwością poradzić sobie z dziesięcioma sekundami, które upływają przed włączeniem rozwiązania zapasowego. Obecnie do przeszłości należą przerwy towarzyszące awariom tego rodzaju.

Połączenie zapasowe w opisanej konfiguracji nie korzysta z monitorowania. Jeśli łącze zapasowe jest obsługiwane przez innego usługodawcę, jest mało prawdopodobne, że obaj usługodawcy będą mieli awarię jednocześnie. W takim wypadku protokół BGP by nas nie uratował. ■

Listing 2: Skrypt działania HA-VPN

```
#!/bin/bash
# HA-VPN -- skrypt działania
#VERBOSE=-v
VERBOSE=""
NAME=`basename $0`
LOG="/usr/bin/logger -t HA-VPN"
PARAMETER_FAULT=0
if [ $# -ne 1 ] ; then
    PARAMETER_FAULT=1
else
    PARAMETER=$1
    case $PARAMETER in
        start) ;;
        stop) ;;
        *) PARAMETER_FAULT=1 ;;
    esac
fi
if [ $PARAMETER_FAULT -ne 0 ] ; then
    $LOG " ${NAME}: called with
:*$*: ==> parameter error, abort"
    echo "`date +%Y%m%d%H%M%S`
${NAME}: parameter error, abort"
    exit 1
fi
ACTION_FAIL_START="/etc/init.d/
/heartbeat stop"
ACTION_OK_AGAIN="/etc/init.d/
heartbeat start"
case $PARAMETER in
    start) $LOG ${ACTION_
FAIL_START} ;
        ${ACTION_FAIL_START} ;;
    stop) $LOG ${ACTION_OK_AGAIN}
;
        ${ACTION_OK_AGAIN} ;;
esac
exit 0
```

INFO

- [1] RFC 2401: <http://www.ietf.org/rfc/rfc2401.txt>
- [2] Freeswan: <http://www.freeswan.ca/code/super-freeswan/>
- [3] RFC 1745: <http://www.ietf.org/rfc/rfc1745.txt>
- [4] SSPE: <http://sspe.sourceforge.net>
- [5] Linux-HA: <http://www.linux-ha.org>

AUTOR

Johannes Hubertz jest fanem Linuksa od 0.99.2, głównie dlatego, że Linux pozwala mu zejść na poziom pojedynczych bitów, kiedy jest mu to potrzebne. Jeśli chodzi o sprawy zawodowe, to Johannes zarządza kilkudziesięcioma komputerami dla jednego z największych przedsiębiorstw informatycznych w Europie od 1996 roku.