Startowanie systemu klastra dla rozwiązania OpenSSI

Projekt z przedmiotu RSO

Sławomir Zielski
S.Zielski@elka.pw.edu.pl

17 czerwca 2005

Spis treści

—

Laboratoryjne rozwiązanie klastrowe wykorzystuje system Linux Debian. Startowanie systemu rozpoczyna się od uruchomienia procesu init, który rozpoczyna od przetwarzania standardowego skryptu /etc/inittab. Skrypt ten następnie uruchamia skrypt /etc/init.d/rc z parametrem 2. Plik /etc/init.d/rc zawiera rozszerzenia systemu Debian o implementacje rozwiązania OpenSSI.

1 Rozszerzenia w pliku: /etc/init.d/rc

Plik ten rozpoczyna swoje działanie od sprawdzenia, czy aktualnie uruchomione jądro zawiera w sobie rozwiązanie OpenSSI (/bin/clusternode_num). Jeżeli tak to następuje ustawienie zmiennych: SSI=y i DBLOCK='/etc/init.d/debian.lock'. Dalszym rozszerzeniem systemu jest sprawdzenie, czy poprzedni poziom pracy systemu był inny niż 'N'. Jeżeli tak wówczas sprawdzane jest czy w katalogu /etc/rc2.d istnieją linki symboliczne których nazwa rozpoczyna się od K[0-9][0-9]. Warunek ten nigdy nie jest spełniony ponieważ cechą dystrybucji Debian jest to że nie posiada linków symbolicznych kończących daną usługę w katalogach /etc/rc$runlevel.d. Skutkiem tego jest tkwiący błąd w linii 109 skryptu: ONSVC='/bin/onsvc –quiet –procview=local $subsys $DBUNLOCK'. Zmienna DBUNLOCK nie istnieje i prawdopodobnie powinna to być zmienna DBLOCK (zgodnie z wyjaśnieniem Aneesh'a Kumar'a). Kolejna cześć skryptu odpowiedzialna jest za uruchomienie usług systemowych dla zadanego poziomu działania (runlevel). Sprawdzane usługi są pobierane z katalogu: /etc/rc$runlevel.d (np.: /etc/rc2.d). Na początku tego fragmentu sprawdzane jest czy na poprzednim poziomie działania dana usługa była uruchomiona i czy na docelowym poziomie działania istnieje skrypt kończący działanie tej usługi. Jeżeli usługa była uruchomiona i nie istnieje skrypt kończący jej działanie na docelowym poziomie, wówczas nie ma potrzeby przeprowadzać

dalszych działań. Wówczas analizowana jest następna usługa systemowa z katalogu /etc/rc$runlevel.d. Jeżeli analizowana usługa systemowa nie była uruchomiana (i nie zakończona) na poprzednim poziomie uruchamiania, wówczas sprawdzane jest czy nie jest ona uaktywniona 'ręcznie' przez administratora systemu. Ściśle sprawdzane jest czy istnieje plik: /cluster/var/lock/subsys/L???$subsys. Jeżeli usługa ta jest uruchomiona to następuje sprawdzenie następnej usługi systemowej z katalogu /etc/rc$runlevel.d. Jeżeli dana usługa nie jest uruchomiona wówczas następuje przygotowanie parametrów jej uruchomienia:

- znajdowana jest klasa węzłów (node_class) uruchamianej usługi. Realizowane jest to poprzez znalezienie w pliku /etc/rc.nodeinfo przestawionego w tabeli 1 analizowanej usługi i pobranie wartosci CLASS z drugiej kolumny.
- następnie następuje dokonywany jest wybór pomiędzy poziomami uruchamiania. Dla poziomów 0 (halt) i 6 (reboot) usługa jest zamykana. Natomiast dla wszystkich innych poziomów analizowana usługa jet uruchamiana. Dla przykładu, dla drugiego poziomu działania polecenie uruchamiające dana usługę będzie miało postać: /bin/onclass all /etc/init.d/debian.lock /etc/rc2.d/devfsd start
- na końcu tej części następuje wywołanie wewnętrznej funkcji set_global_lock z parametrem nazwy uruchamianej funkcji systemowej. Zadaniem tej funkcji jest aktualizacja licznika uruchomionych usług systemowych dotyczących klastra: /cluster/var/lock/subsys/counter.

Funkcja ta sprawdza przekazany parametr ze względu na to czy nie posiada jednej z trzech wartości: umountfs, halt lub reboot, dla których zwiększanie wartości licznika nie ma sensu. Plik /etc/init.d/debian.lock pełni rolę usługową dla skryptu /etc/init.d/rc. Jego zadaniem jest zamiana linku symbolicznego na nazwę własnego skryptu oraz w zależności od parametru start lub stop uruchomienie lub zatrzymanie danej usługi.

- network all Y
- random all Y
- portmap all Y
- nfslock all Y
- xinetd all Y
- inetd all Y
- inetutils-inetd all Y
- rawdevices all Y
- keytable all Y
- ntpd all Y
- syslog all Y
- ssh all Y

- umountfs initnode N
- urandom all Y
- umountnfs.sh all Y
- sysklogd all Y
- klogd all Y
- rmnologin all Y
- ha-lvs all Y
- devfsd all Y
- stop-bootlogd all Y
- makedev all Y
- sendsigs all Y
- halt initnode N
- reboot initnode N
- single initnode N
- loadlevel all Y
- dhcp initnode Y
- atftpd initnode Y
- networking all Y
- SSIfailover all N
- ssi-ksync all M
- ssi-ksync all Y
- ntpdate initnode Y
- ifupdown initnode N
- ifupdown-clean initnode N
- gdm initnode N

Tab.1. Tablica usług systemowych i sposobu ich uruchamiania (na wszystkich węzłach lub tylko na węźle inicjującym). Zawartość pliku: /etc/rc.nodeinfo.

2 Plik konfiguracyjny dla nowo przyłączanego węzła /etc/init.d/rc.nodeup

Skrypt /etc/init.d/rc.nodeup jest uruchamiany gdy węzeł jest podłączany do klastra na tym węźle. Skrypt ten rozpoczyna swoje działanie od uruchomienia pliku /etc/init.d/rcSSI, który odpowiedzialny jest za uruchomienie wszystkich startowych skryptów z katalogu /etc/rcSSI.d/, nie będących linkami symbolicznymi:

Skrypt /etc/init.d/rc.nodeup jest uruchamiany gdy węzeł jest podłączany do klastra na tym węźle. Skrypt ten rozpoczyna swoje działanie od uruchomienia pliku /etc/init.d/rcSSI, który odpowiedzialny jest za uruchomienie wszystkich startowych skryptów z katalogu /etc/rcSSI.d/, nie będących linkami symbolicznymi:

- S01devfsd - skrypt odpowiedzialny za uruchomienie demona zarządzającego systemowymi urządzeniami
- plikowymi;zapewnia prawidłowe prawa dostępu oraz udostępnia linki symboliczne urządzeń;
- S10checkroot.sh - sprawdzenie korzenia systemu plików;
- S18hwclockfirst.sh - ustawienie daty i zegara systemowego na węźle inicjującym;
- S30checkfs.sh - sprawdzenie wszystkich systemów plików;
- S35mountall.sh - montowanie systemu plików;
- S55bootmisc.sh - zabezpiecza system przed zalogowaniem się do niego przed zakończeniem startowania systemu; wspiera rejestrowanie logowania się do klastra; zachowuje wiadomości jądra w /var/log/dmesg;

Poza tym w powyższym katalogu znajdują się następujące linki symboliczne:

- S02mountvirtfs - skrypt odpowiedzialny za zamontowanie wszystkich wirtualnych systemów plików dostarczanych przez jądro oraz wymaganych domyślnie;
- S05bootlogd - odpowiedzialny za rozpoczęcie lub zakończenie programu tworzącego dziennik bootlogd;
- S05initrd-tools.sh - odmontowuje i zwalnia initrd;
- S05keymap.sh - ustawia kodowanie znaków;
- S20modutils - ładowanie odpowiednich modułów;
- S30procps.sh - ustawienie zmiennych jądra z pliku /etc/sysctl.conf
- S36discover - uruchomienie programu /sbin/discover odpowiedzialnego za roz
- S39ifupdown - obsługa interfejsów sieciowych;
- S40hostname.sh - przypisanie nazwy hostowi;
- S40hotplug - uruchamia i stopuje każdy wymienny podsystem;
- S40networking - zarządza interfejsami sieciowymi i konfiguruje ich opcje;

- S41ha-lvs - uruchamia skrypt dla SSI Ha-LVS;
- S45mountnfs.sh - uruchamia mapowanie portów, w przypadku gdy jest to możliwe; montuje system plików NFS;
- S55bootmisc.sh - zabezpiecza system przed zalogowaniem się do niego przed zakończeniem startowania systemu; wspiera rejestrowanie logowania się do klastra; zachowuje wiadomości jądra w /var/log/dmesg;
- S55urandom - zachowuje psełdolosowe ziarno pomiedzy kolejnymi uruchomieniami komputera.

Następnie następuje wykonanie usług systemowych umieszczonych w katalogu: /cluster/var/lock/subsys i jednocześnie umieszczonych w pliku /etc/rc.nodeinfo.

L000devfsd - jest to prawie ten sam skrypt, który był uruchamiany z katalogu /etc/rcSSI.d. Różnice są dwie:

- w pliku S01devfsd było:
  #SSI_XXX Hack!!!
  #[-d /proc/1] -o mount -n /proc
  cmount,
  natomiast w pliku omawianym jest:
  [-d /proc/1] -o mount -n /proc
- w pliku S01devfsd było:
  PATH=/bin:/usr/bin:/sbin:/usr/sbin.
  natomiast w pliku omawianym jest:
  PATH=/bin:/sbin.

Pierwsza zmiana oznacza rozszerzenie możliwości montowania o CSDL (Context Dependent Symbolic Links) w skryptach /etc/rcSSI.d:

- 001sysklogd - startuje demona tworzenia rejestrów dziennikowych. Demon ten jest odpowiedzialny za rejestrowanie wiadomości otrzymywanych od programów i mechanizmów zarówno lokalnych jak zdalnych.
- L002klogd - startuje demona rejestrów dziennikowych jądra.
- L003portmap - uruchamia i zatrzymuje demona mapowania portów.
- L004atftpd - skrypt służący do uruchomienia serwera atftpd. Uruchamiany tylko na węźle inicjującym.
- L005dhcp - uruchomienie serwera dhcp. Uruchamiany tylko na węźle inicjującym.
- L006inetd - słuzący do uruchomienia serwera initd.
- L007makedev - tworzy plik urządzenia
- L008ssh - uruchamia i zatrzymuje demona ssh.
- L009SSIfailover - zawiera mechanizmy omijania awarii węzła głównego dla węzłów zależnych.

- L010loadlevel - skrypt startujący dla klastra mosix równoważącego obciążenie.
- L011gdm - zarządza systemem wyświetlania GNOME. Uruchamiany tylko na węźle inicjującym.
- L012rmnologin - usuwa skrypt usuwający blokadę zapisu do systemu.
- L013stop-bootlogd - zatrzymuje program tworzący rejestr dziennikowy uruchamiania systemu.

Dalsza część skryptu ustawia węzeł w stan aktywny. Następnie ustawiana jest ścieżka route'owania do węzła głównego. Na końcu tego skryptu są ładowane moduły biblioteczne dla obsługi pendrive'a i SCSI.

Literatura

[www] www.openssi.org.