Dydaktyka Informatyki 10(2015)
ISSN 2083-3156
DOI: 10.15584/di.2015.10.18
http://www.di.univ.rzeszow.pl

Wydział
Matematyczno-Przyrodniczy UR
Laboratorium Zagadnień
Społeczeństwa Informacyjnego

Jacek WOŁOSZYN

Dr inż., Uniwersytet Technologiczno-Humanistyczny w Radomiu, Wydział Informatyki i Matematyki, Katedra Informatyki, ul. Malczewskiego 29, 26-600 Radom; jacek@delta.pl

AUTOMATYZACJA PROCESU ARCHIWIZACJI PRZYROSTOWEJ DANYCH Z WYKORZYSTANIEM GIT

AUTOMATING THE PROCESS OF INCREMENTAL BACKUP DATA USING GIT

Słowa kluczowe: bezpieczeństwo, archiwizacja, automatyzowanie procesów.
Keywords: system security, firewall, identification, encryption.

Streszczenie

Wykonywanie przyrostowych kopii zapasowych jest zdecydowanie bardziej efektywne od tworzenia archiwizacji całościowej danych. W przypadku dużych zbiorów różnice w czasie procesu wykonywania kopii i objętości pliku tworzonego materiału widać natychmiast, ponieważ kopiowane są tyko pliki, w których nastąpiły zmiany, co znacznie zmniejsza ilość miejsca potrzebnego na przechowywanie kolejnych kopii.

W artykule tym pokazano przykład wykorzystania systemu do wykonywania kopii zapasowej z kontrolą wersji backupu, która pozwala w przypadku częstych zmian treści w plikach szybko odnaleźć i przywrócić właściwą wersję. Połączenie tego rozwiązania z Cronem umożliwia całkowitą automatyzację procesu.

Summary

This article shows an example of use of the system to perform a backup of the backup version control, which allows for frequent changes to the content in the files quickly locate and restore the correct version. Combining this approach with Cron allows to completely automate the process.

Wprowadzenie

Świadomość wykonywania kopii zapasowych jest oczywista. Każdy o tym wie, a mimo to prawdopodobnie każdy z nas miał sytuację, w której w jakiś sposób stracił dane i dopiero pomyślał: *szkoda, że nie zrobiłem kopii zapasowej*. Ale wtedy już jest za późno.

Poniższy artykuł proponuje pewne rozwiązanie, które można zastosować w przypadku tworzenia projektu, gdzie jest często wymagany powrót do różnych wcześniejszych punktów pracy. Zdarza się to często podczas tworzenia projektów, kiedy przyjęta ścieżka postępowania okazuje się niewłaściwa i należy powrócić do wcześniejszych rozważań, aby wybrać inny kierunek drogi.

1. Przygotowanie środowiska

Należy wykonać kopię zapasową katalogu z plikami źródłowymi tworzonego oprogramowania.

Wraz z rozwojem oprogramowania i pisaniem kolejnych procedur kodu źródłowego zmianie ulegają poszczególne pliki źródłowe systemu. Po wykonaniu kopii całościowej kolejne kopie bezpieczeństwa powinny być wykonywane przyrostowo uwzględniając tylko pliki, które uległy modyfikacji.

Ścieżka do katalogu źródłowego: /root/dv

Jest to katalog, który będzie archiwizowany. Aby do niego wejść należy wydać polecenie:

```
root@dlt:~/dj# cd /root/dv
```

Dla większego bezpieczeństwa[1] przechowywania danych ta sama kopia będzie przechowywana na komputerze zdalnym o adresie IP:111.111.111.111.

W tym celu utworzony zostanie docelowy katalog, w którym będą one umieszczane.

Operacji tej można dokonać bezpośrednio na maszynie zdalnej o ile mamy do niej fizyczny dostęp lub też zdalnie z wykorzystaniem protokołu ssh[2].

```
root@dlt:~/dj# ssh root@111.111.111.111
```

Po uzyskaniu wymaganych praw dostępu do zdalnego komputera należy utworzyć opisany wcześniej katalog.

```
root@dot:~# mkdir -p /root/kopie/kopie.git
root@dot:~# cd /root/kopie/kopie.git/
root@dot:~/kopie/kopie.git# git init --bare
Initialized empty Git repository in /root/kopie/kopie.git/
```

W wyniku przeprowadzonych operacji została zainicjalizowana pusta struktura katalogów do przechowywania kopii zapasowych. Jego przekrój wygląda następująco:

[1] G. Stepanek, *Software Project Secrets*, Apress 2012.
[2] Ch. McNab, *Network Security Assessment*, O'Reilly 2007.

```
root@dot:~/kopie/kopie.git# ls -li
razem 32
44433651 drwxr-xr-x 2 root root 4096 sty 15 12:41 branches
44433653 -rw-r--r-- 1 root root   66 sty 15 12:41 config
44433648 -rw-r--r-- 1 root root   73 sty 15 12:41 description
44433652 -rw-r--r-- 1 root root   23 sty 15 12:41 HEAD
44433639 drwxr-xr-x 2 root root 4096 sty 15 12:41 hooks
44433649 drwxr-xr-x 2 root root 4096 sty 15 12:41 info
44433654 drwxr-xr-x 4 root root 4096 sty 15 12:41 objects
44433636 drwxr-xr-x 4 root root 4096 sty 15 12:41 refs
```

Po wylogowaniu się z maszyny zdalnej[3] powracamy do katalogu z plikami źródłowymi, aby dodać szczegóły dotyczące użytkownika. W tym przypadku można uzupełnić system o dane nazwy użytkownika i jego adres mailowy. W przypadku pracy nad projektem zespołowym ułatwi to filtrowanie kopii w procesie szukania właściwych plików.

```
root@dlt:~/dv# git config --global user.name "Jacek"
root@dlt:~/dv# git config --global user.email jacek@jacek.pl
```

Po tej operacji z poziomu katalogu, w którym znajdują się pliki do archiwizacji należy zainicjować katalog źródłowy hosta, dla którego ma zostać sporządzona kopia.

```
root@dlt:~/dv# git init
Initialized empty Git repository in /root/dv/.git/

root@dlt:~/dv# git commit --allow-empty -am "init"
[master (root-commit) 691f655] init
```

Nie opuszczając katalogu źródłowego należy dodać katalog zdalny systemu Git oraz zsynchronizować kopię zapasową.

```
root@dlt:~/dv# git remote add origin root@111.111.111.111:/root/kopie/kopie.git
root@dlt:~/dv# git push origin master
Enter passphrase for key '/root/.ssh/id_rsa':
Counting objects: 2, done.
Writing objects: 100% (2/2), 152 bytes, done.
Total 2 (delta 0), reused 0 (delta 0)
To root@111.111.111.111:/root/kopie/kopie.git
* [new branch]      master -> master
```

[3] R. Pinkal Pollei, *Debian 7 System Administration Best Practices*, Packt 2013.

2. Wybór danych do archiwizacji i automatyzacja procesu

Jeżeli nie jest wymagane śledzenie wszystkich plików w katalogu źródłowym, a jedynie wybranych jego elementów, to można za pomocą polecenia add odpowiednio zmodyfikować wykonywanie kopii. Wykonanie polecenia add * dodaje do listy archiwizowania wszystkie katalogi i pliki. Można dodać tylko wybrane używając polecenia w następujący sposób git add .doc, git add *.txt. W podobny sposób można usunąć z listy wybrane pliki i katalogi za pomocą polecenia git rm nazwa_pliku lub wykonać to masowo git rm *.txt.

```
root@dlt:~/dv# git add *
root@dlt:~/dv# git commit -am "accept - $(date)"
[master da6d275] accept - czw, 15 sty 2015, 13:03:16 CET
 1 file changed, 41 insertions(+)
 create mode 100644 __init__.py
 create mode 100644 dv.e4p
root@dlt:~/dv# git push
Enter passphrase for key '/root/.ssh/id_rsa':
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 757 bytes, done.
Total 4 (delta 0), reused 0 (delta 0)
To root@111.111.111.111:/root/kopie/kopie.git
   691f655..da6d275  master -> master
```

Umieszczenie odpowiedniego skryptu w pliku Crontab'a[4] pozwoli na automatyzację procesu z odpowiadającymi nam parametrami częstotliwości wykonywania operacji.

Dla przypomnienia kolejność wpisu w pliku crontab oznacza w kolejności minuta /0–59/, godzina /0–23/, dzień miesiąca /1–31/, miesiąc /1–12/, dzień tygodnia /0–7/ /niedziela=0, poniedziałek=1 itd./
Przykładowy wpis może wyglądać następująco:

```
0 * * * * skrypt uruchamiający
```

Zapewni on nam uruchomienie procesu archiwizacji o każdej pełnej godzinie. Odpowiednia modyfikacja tych parametrów pozwala na dowolne ustawienia uruchomienia skryptu.

[4] T. Bauts, T. Dawson, G. Purty, *Linux Network Administrator*, O'Reilly 2005; Ch. Negus, *Linux. Biblia. Ubuntu, Fedora, Debian i 15 innych dystrybucji*, Helion Gliwice 2012; E. Nemeth, G. Snyder, R. Trent, H. Whaley, *Ben Unix and Linux system administration handbook fourth edition*, Pretence Hall 2010.

3. Odtwarzanie danych

Przeglądu wszystkich utworzonych kopii zapasowych można dokonać wydając polecenie:

```
root@dlt:~/dv# git log
commit da6d2752cac1b9754917b7825db0f402a2e2f8ae
Author: Jacek <jacek@jacek.pl>
Date:   Thu Jan 15 13:03:16 2015 +0100

    accept - czw, 15 sty 2015, 13:03:16 CET

commit 691f6554c33125dfd3dadf25361a5a19447637d5
Author: Jacek <jacek@jacek.pl>
Date:   Thu Jan 15 12:52:49 2015 +0100

    init
```

Przywrócenie bieżącego katalogu roboczego przy użyciu kopii zapasowej jest możliwe przy użyciu polecenia:

```
root@dlt:~/dv# git checkout  da6d2752cac1b9754917b7825db0f402a2e2f8ae
Note: checking out 'da6d2752cac1b9754917b7825db0f402a2e2f8ae'.

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using -b with the checkout command again. Example:

  git checkout -b new_branch_name

HEAD is now at da6d275... accept - czw, 15 sty 2015, 13:03:16 CET

root@dlt:~/dv# git commit -am "odt - @ $(date) accept:
da6d2752cac1b9754917b7825db0f402a2e2f8ae"

root@dlt:~/dv# git push
Enter passphrase for key '/root/.ssh/id_rsa':
Everything up-to-date
```

Jeśli lokalny katalog z kopią zapasową został z jakiegoś powodu uszkodzony należy wówczas skorzystać z katalogu z kopią zapisanego na maszynie zdalnej.

```
root@dlt:~/dv# git clone root@111.111.111.111:/root/kopie/kopie.git
Cloning into 'kopie'...
Enter passphrase for key '/root/.ssh/id_rsa':
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 6 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (6/6), done.
```

Wnioski

Każdego użytkownika komputera spotkała zapewne chociaż raz niemiła sytuacja, kiedy to w wyniku popełnionego błędu, awarii sprzętu, czy innych zdarzeń losowych stracił dane, nad którymi ciężko pracował pisząc książkę, artykuł czy kolejne linie kodu aplikacji. Jeśli na bieżąco nie wykonywał kopii zapasowej stracił godzinę, pięć godzin, dzień, a czasami i całą swoją pracę.

Przedstawione powyżej rozwiązanie pozwala uniknąć takiej sytuacji. W zależności od parametrów konfiguracji, jakie zostaną przyjęte kopia zostaje wykonana automatycznie, oznaczona znacznikami czasu, autora i umieszczona w dwóch niezależnych miejscach na komputerze lokalnym, jak i zdalnym w określonym miejscu. Zaprezentowane rozwiązanie szczególnie sprawdza się w przypadku budowania rozbudowanych aplikacji, w sytuacji, kiedy należy wrócić do określonego wcześniej stanu projektu, celem poszukania innej ścieżki rozwiązań.

Bibliografia

Bauts T., Dawson T., Purty G., *Linux Network Administrator*, O'Reilly 2005.

McNab Ch., *Network Security Assessment*, O'Reilly 2007.

Negus Ch., *Linux. Biblia. Ubuntu, Fedora, Debian i 15 innych dystrybucji*, Helion, Gliwice 2012.

Nemeth E., Snyder G., Trent R., Whaley H., *Ben Unix and Linux system administration handbook fourth edition*, Pretence Hall 2010.

Pollei Pinkal R., *Debian 7 System Administration Best Practices*, Packt 2013.

Stepanek G., *Software Project Secrets*, Apress 2012.