Конфигурационные файлы для вашего скрипта

Общая информация

Для этой задачи вам не нужно писать большие процедуры синтаксического анализа (если вы не хотите, чтобы это было на 100% безопасно или вам нужен специальный синтаксис файла) - вы можете использовать команду Bash source . Исходный файл должен быть отформатирован в формате key="value", в противном случае bash попытается интерпретировать команды:

```
#!/bin/bash
echo "Чтение конфигурации ...." >&2
источник /etc/cool.cfg
echo "Конфигурация для имени пользователя: $ cool_username" > & 2
echo "Конфигурация для целевого хоста: $ cool_host"> & 2
```

Итак, откуда берутся эти переменные? Если все работает нормально, они определены в файле /etc/cool.cfg, который является исходным кодом для текущего скрипта или оболочки. Примечание: это **не** то же самое, что выполнить этот файл как скрипт! Исходный файл, скорее всего, содержит что-то вроде:

```
cool_username="гость"
cool_host="foo.example.com "
```

Это обычные инструкции, понятные Bash, ничего особенного. Конечно (и это большой недостаток при обычных обстоятельствах) исходный файл может содержать **все**, что понимает Bash, включая вредоносный код!

`source` команда также доступна под именем `.` (точка). Использование точки идентично:

```
#!/bin/bash
echo "Чтение конфигурации...." >&2
. / etc/cool.cfg #обратите внимание на пробел между точкой и косой че
ртой /etc.cfg
echo "Конфигурация для имени пользователя: $ cool_username" > & 2
echo "Конфигурация для целевого хоста: $ cool_host"> & 2
```

Конфигурации для каждого пользователя

Существует также способ предоставить общесистемный конфигурационный файл в /etc и пользовательскую конфигурацию в ~/(домашняя страница пользователя), чтобы переопределить общесистемные значения по умолчанию. В следующем примере конструкция if/then используется для проверки наличия конфигурации, зависящей от пользователя:

```
#!/bin/bash
echo "Чтение общесистемной конфигурации...." >&2
. / etc/cool.cfg
если [ -r ~/.coolrc ]; затем
повторите "Чтение конфигурации пользователя ...." > &2
 . ~/.coolrc
fi
```

Защитите его

Как упоминалось ранее, исходный файл может содержать все, что может скрипт Bash. По сути, это включенный скрипт Bash. Это создает проблемы с безопасностью. Злоумышленник может "выполнить" произвольный код, когда ваш скрипт использует свой конфигурационный файл. Возможно, вы захотите разрешить только конструкции в форме `NAME=VALUE` в этом файле (синтаксис присваивания переменных) и, возможно, комментарии (хотя технически комментарии не важны). Представьте себе следующий "конфигурационный файл", содержащий некоторый вредоносный код:

```
# классный конфигурационный файл для моего еще более крутого скрипта
Имя пользователя = god_only_knows
hostname=www.example.com
пароль = секретный; echo rm -rf ~/*
параметр=foobar && echo "Вы выиграли pwned!";
# эй, смотри, далее следует странный код...
эхо "Я вирус черепа ..."
echo rm -fr ~/*
mailto=netadmin@example.com
```

Вы не хотите, чтобы эти `echo` -команды (которые могут быть любыми другими командами!) выполнялись. Один из способов быть немного безопаснее - фильтровать только те конструкции, которые вы хотите, записывать отфильтрованные результаты в новый файл и создавать новый файл. Мы также должны быть уверены, что в конец одного из наших параметров name=value не было добавлено что-то гнусное, возможно, с использованием разделителей команд; или && . В таких случаях, возможно, проще всего просто полностью игнорировать строку. `grep -E` Здесь нам поможет Egrep (), он фильтрует по описанию:

```
#!/bin/bash
configfile='/etc/cool.cfg'
configfile_secured='/tmp/cool.cfg'

# проверьте, содержит ли файл что-то, что нам не нужно
, если egrep -q -v '^#|^[^ ]*=[^;]*' "$ configfile"; затем
повторите "Файл конфигурации нечист, очистите его ..."> &2
 # фильтровать оригинал в новый файл
egrep '^#|^[^ ]*=[^;&]*' "$ configfile" > "$configfile_secured"
configfile="$configfile_secured"
fi

# теперь используйте исходный или отфильтрованный вариант
исходного кода "$configfile"
```

Чтобы было понятно, что он делает: egrep проверяет, содержит ли файл что-то, что нам не нужно, если да, egrep фильтрует файл и записывает отфильтрованное содержимое в новый файл. Если это сделано, исходное имя файла изменяется на имя, сохраненное в переменной `configfile` . Файл, названный этой переменной, является исходным, как если бы это был исходный файл.

Этот фильтр разрешает только `NAME=VALUE` комментарии и в файле, но не запрещает все методы выполнения кода. Я рассмотрю это позже.

Обсуждение

Iñigo (http://poisonbit.wordpress.com), 2010/09/05 14:55 ()

Пожалуйста, примите во внимание $()

```
 inigo@crono:~/tmp/cfg$ cat test.cfg
name=fooo
address="fooo@fooo.org $(echo $(ls))"
```

Использование фильтра статей:

```
 inigo@crono:~/tmp/cfg$ bash cfg.sh имя
пользователя: fooo
адрес: fooo@fooo.org cfg.sh тест.cfg
```

$(echo $(ls)) (или любая другая команда) может быть выполнена.

То же самое можно сделать с помощью `...`

Ян Шампера, 2010/09/13 04:34 ()

Да, вы абсолютно правы.

Я хотел бы знать, прежде чем вносить что-то еще в код, имеет ли смысл перевести эту статью в автономный режим. Способ хорош, прост, эффективен, но непригоден, что бы вы ни делали. Это и остается обходным путем, чтобы не кодировать слишком много для файловой системы конфигурации.

Рейнир Бун, 2011/01/19 15:40 ()

Я успешно использовал конструкцию, подобную

```
 MAIL_TO=`perl -ne'/^\s*MAIL_TO\s*=\s*([\s\w@\d,\._-]+)/ && do {p
rint $1; exit;}' $CONFIG_FILE`
```

для чтения пользователей, которым нужно отправлять почту, из файла конфигурации...

Это немного зависит от того, что именно будет в вашей переменной конфигурации, возможно, вам придется настроить регулярное выражение.

Амин МЕЧИФИ (http://www.linuxhope.com), 2011/11/26 11:03 ()

** Пожалуйста, не удаляйте этот совет. Это хороший скрипт!

С точки зрения безопасности, если кто-то может работать с файлом конфигурации, он также сможет редактировать файл сценария оболочки в первую очередь.

Если вы действительно обеспокоены безопасностью, вы можете сделать следующее:

- как только вы убедитесь, что ваш конфигурационный файл завершен, выполните md5sum и жестко запрограммируйте его в сценарии оболочки. Таким образом, скрипт будет проверять md5sum файла conf каждый раз, когда он его вызывает. - Проблема в том, что если вы что-то измените в файле conf, вам нужно выдать новый md5sum и поместить в оболочку. - Вы можете быть более агрессивным (параноиком?), Но, разделив конфигурационный файл на 2 или более файлов, проверьте md5sum каждого из них. Затем вы также тестируете md5sum всего файла и сравниваете все. Это всего лишь условие для случая, когда кто-то создает измененный конфигурационный файл, но с тем же md5sum (теоретически возможно).

Ян Шампера, 2011/11/29 06:54 ()

Злоумышленник, который может редактировать конфигурацию, не обязательно может редактировать сценарий, просто подумайте о конфигурациях в dot-file в домашнем каталоге или тому подобном.

Вы также правы, да, но предупреждение здесь абсолютно необходимо

ano (http://www.ano.de), 2012/07/27 22:50 ()

есть много способов пробиться через фильтр

jhv@hyperion:~/Desktop/test$ cat conf #это конфигурационный файл test1=ssss && коснитесь yougotowned0 test2=aaa; коснитесь yougotowned1 test2=`./virus` ./virus;k= www 1test2=false||коснитесь yougotowned2 test2=false коснитесь yougotowned3 test3=$(./virus) ./virus\= a gg=./virus $gg;h=hh

jhv@hyperion:~/Desktop/test$ ls -l insgesamt 20 -rwxr-xr-x 1 jhv jhv 38 2012-07-28 00:26 очистить -rwxr-xr-x 1 jhv jhv 233 2012-07-28 00:42 conf -rwxr-xr-x 1 jhv jhv 449 2012-07-28 00:40 config_loader-rwxr-xr-x 1 jhv jhv 535 2012-07-28 00:22 org -rwxr-xr-x 1 jhv jhv 45 2012-07-28 00:22 вирус lrwxrwxrwx 1 jhv jhv 28 2012-07-28 00:29 вирус =a → /home/jhv/Desktop/test/вирус jhv@hyperion: ~/Desktop/test $ #это "вирус" jhv @hyperion:~/ Desktop/ test $ cat virus #!/bin/ bash echo boom > и 2 касания, принадлежащие СЛУЧАЙНОМУ jhv @hyperion:~/ Desktop/test $ ./config_loader Файл конфигурации нечист, его очистка ... бум-бум/tmp/cool.cfg: Zeile 6: 1test2=false: команда nicht gefunden. boom boom boom jhv@hyperion:~/Desktop/test$ ls очистить организацию, владеющую 14191 вирусом, который вам известен 1 conf, владеющую 10656, владеющую 15845 вирусом = a, который вам известен 2 config_loader, владеющую 10851, владеющую 5682, который вам известен 0, который вам известен 3 jhv@hyperion:~/Desktop /тест$

Фахми М.Ф., 08.02.2012 06:52 ()

Привет, мне нужно настроить сборку для каждой тестовой среды с определенным набором файлов в сборке, относящимся к конфигурации конкретной среды, например, как показано ниже

Имя файла - audit.properties

PROVIDER_URL=http://10.10.10.10:8080,11.11.11.11:8082 (http://10.10.10.10:8080,11.11.11.11:8082)

database.username=имя пользователя database.psswd=пароль

В настоящее время я настраиваю вручную сохранение в документе всех сохраненных конфигураций в тестовой среде и конфигурации перед развертыванием сборки. Что мне нужно, так это получить сценарий оболочки для настройки свойств / конфигурации, считывающий конкретную конфигурацию среды из определенного файла, который содержит, какой файл необходимо настроить и какие свойства необходимо настроить, и какое значение нужно ввести. тогда его также было бы легко поддерживать, поскольку эти значения часто меняются, пожалуйста, помогите мне в этом

Заранее благодарю вас, Faz m Fah

Джо (http://sourceforge.net/projects/duplexpr/), 2013/01/18 06:24 (), 2014/10/06 04:30 ()

Это вариант описанных выше методов, но с использованием sed, над которым я сейчас работаю. Это далеко не пуленепробиваемый, но это может дать вам некоторые идеи:

```
 ## Удалить все комментарии, нулевые и пустые строки
 ## Удалите все строки, содержащие в них гадости оболочки
 ## (не будет перехватывать переведенные значения, такие как 0x2
4)
 ## Затем передавайте только те строки, которые начинаются с допу
стимого имени параметра
 ## в рабочий файл
 < "${PARAM_FILE}" sed -rn -e '/^#/d
 /^ /d
 /^$/d
 /[$`;><{}%|&!]/ d
 /^DEST_DIR=|^DEST_LABEL=|^DRYRUN=|^MOUNT1=|^MOUNT2=|^NAME=|^PRUN
E=|^SOURCE_DIR=|^SOURCE_LABEL=|^TASK_NAME=|^UNMOUNT_SOURCE=/p'
> "$ {PARAM_WORK}"

... еще несколько правок ...

источник "$ {PARAM_WORK}"
```

Том, 2013/02/19 13:27 ()

Если вы хотите сделать его немного более безопасным, задайте для конфигурационного файла разрешение 600, а владельца - исполнителю скрипта. Затем проверьте в своем скрипте, все ли в порядке с владельцем и разрешением. Все еще не идеально, я знаю

Код наугад (http://codeatrandom.blogspot.com/), 2013/05/15 03:25 (), 2014/10/06 04:29 ()

Спасибо за отличную статью, она очень помогла мне написать мой скрипт! Если я действительно беспокоюсь о безопасности, что я не всегда делаю с частными скриптами, я немного более параноидален, и описанных выше подходов для меня недостаточно, поскольку они нарушают две очень фундаментальные концепции безопасности:

- не используйте черные списки, используйте белые списки (вы никогда не можете быть уверены, что уловили все возможности) - не пытайтесь исправить вредоносный ввод, вместо этого прервите и вызовите ошибку (в противном случае может быть больший вред, и если происходит что-то подозрительное, вы должны быть уведомлены об этом)

(Конечно, для этого правила есть исключения, но не в этом примере.)

Мой подход к этому таков:

```
#!/bin/bash

c_file=config_file_test

неизвестно=`cat $c_file | grep -Evi "^(#.*|[a-z]*='[a-z0-9]*')
$"`
если [ -n "$unknown" ]; затем
ошибка echo "в файле конфигурации. Недопустимые строки: "
 echo $неизвестный
выход 1
fi
источник $ c_file
echo "Файл загружен"
```

Это тихая параноидальная версия, позволяющая только строки, начинающиеся с #, в качестве комментариев и объявления переменных типа "hello ='world 88'". Если в вашем конфигурационном файле есть такие вещи, как URL, вы можете добавить необходимые символы (:/?#&) в регулярное выражение. Из строки не должно быть выхода (или я что-то упустил?). Этот способ также имеет то преимущество, что ему не нужен временный файл.

Надеюсь, это поможет кому-нибудь 🙂

Сират, 2013/07/14 09:42 (), 2014/10/06 04:31 ()

Как насчет использования вместо этого белого списка и отклонения файла, если что-то окажется не так:

```
#!/bin/bash

CONFIG_PATH='./bashconfig.conf'
# допустимы строки с комментариями, пустые строки и строки из cho
ose_ANYNAME='any.:Value'
CONFIG_SYNTAX="^\s*#|^\s*$|^[a-zA-Z_]+='[^']*'$"

# проверьте, содержит ли файл что-то, что нам не нужно
, если egrep -q -v "${CONFIG_SYNTAX}" "$CONFIG_PATH"; затем
echo "Ошибка синтаксического анализа файла конфигурации $ {CONFIG
_PATH}". > & 2
echo "Следующие строки в конфигурационном файле не соответствуют
синтаксису:" >& 2
 egrep -vn "${CONFIG_SYNTAX}" "$CONFIG_PATH"
 выход 5
fi

# в противном случае продолжайте и создайте его:
source "$ {CONFIG_PATH}"
```

Джимми, 2014/04/07 19:53 (), 2014/10/06 04:32 ()

Как насчет этого?

```
cat /dev/null > /tmp/rc.conf~

sed '/^ *#/d;/^[:пробел:]*$/d;s/=/ /;' < "/etc/rc.conf" | при чте
нии ключа val
делать
 #проверьте здесь
 #удалить все ненужные
значения = $(echo "$val" | sed 's/[^[:alnum:]\.]//g')
 str="$key=\"$val \""
 echo "$str" >> /tmp/rc.conf~
готово

. /tmp/rc.conf~
```

Это должно удалить самые злые вещи…

Michael Grünewald (https://plus.google.com/u/0/+MichaelLeBarbierGrünewald/about), 2015/06/05 12:41 ()

Как описано в статье моего блога Конфигурационные файлы для сценариев оболочки (http://unix-workstation.blogspot.de/2015/06/configuration-files-for-shell-scripts.html), легко использовать сценарий sed для преобразования файла конфигурации в стиле INI в табличный формат, который можно обрабатывать, например, с помощью awk или read .

howto/conffile.txt Last modified: 2015/08/08 16:00 by bill_thomson