Comma-shell, interaktivní debugger shellu

Tomáš Nesrovnal

Fakulta informačních technologií
České vysoké učení technické v Praze
Obor: Systémové programování
Vedoucí práce: Ing. Jan Baier

12. června 2017

Zadání práce

- Proveďte rešerši existujících nástrojů pro statickou analýzu, krokování a hledání chyb v BASH skriptech.
- Navrhněte a implementujte nástroj, který umožní psát uživatelské skripty pro analýzu příkazů a ovlivňování jejich spouštění a vykonávání. Nástroj musí umožňovat krokovat složitější skripty po jednotlivých příkazech. Pro analýzu spouštěných skriptů využijte vhodný nástroj z rešeršní části.
- Vytvořte ukázkové skripty pro zvrácení, či zamezení efektů základních příkazů z GNU coreutils. Zaznamenávejte jejich spouštění a umožněte jimi provedené změny vrátit do původního stavu.

Cíl práce

- Vytvořit nástroj usnadňující výuku v shellu Bash
- Analýza příkazů před jejich spuštěním
- Jednoduchý, interaktivní debugger příkazů v příkazové řádce
- Sada bezpečných příkazů

Existující řešení

- Debugování skriptů: Bash Debugger (podobné GNU GDB)
- Statická analýza skriptů: ShellCheck

Comma-shell: Hooks

Využití

- Framework pro vytváření uživatelských skriptů
- Spouštění skriptů před vykonáním příkazů z příkazové řádky
- Příkazy k vykonání je možné měnit, nebo nevykonávat

Implementace

- Zastavení všech příkazů pomocí DEBUG trap v módu extdebug
- Příkaz k vykonání se zjišťuje pomocí příkazu history
- Vykonávání příkazů probíhá pomocí příkazu eval
- Hook dostává v parametru příkaz a identifikátor

Ukázka: ShellCheck Hook

```
n@t:~$ printf ”$test” | grep '*test*'
,: ShellCheck:
printf ”$test” | grep '*test*'
       ^-- SC1015: This is a unicode double quote. Delete and retype it.
       ^-- SC2059: Don't use variables in the printf format string. Use printf "..%s.." "$foo".
             ^-- SC1015: This is a unicode double quote. Delete and retype it.
                         ^-- SC2063: Grep uses regex, but this looks like a glob.
,: Now what? [r]un, [s]top, [i]gnore, [p]retype:
```

Comma-shell: Debugger

Využití

- Spouštění částí příkazů v příkazové řádce

Implementace

- Jedná se o hook, původní příkaz se nevykoná
- Pomocí knihovny bashlex se prochází AST příkazu
- K částem příkazu se přidá další kód a v Bashi je vykonán pomocí příkazu eval

Ukázka: Comma-shell Debugger

```
n@t:~$ echo "the year is $(date %+Y)" | grep 2017
,: commash debugger:

echo "the year is $(date %+Y)" | grep 2017
                  ^-- [1] show substituion: $(date %+Y)
                                ^-- [2] show pipe flow: echo "the year is $(date %+Y)"

,: Select your option, [0] debug whole cmd, [q]uit, [r]un normally
```

Comma-shell: Safe commands

Využití

- Wrapper základních příkazů
- Výpis dalších informací
- Změny provedené těmito příkazy lze vrátit zpět

Implementace

- Funkce se stejným jménem jako příkaz z GNU coreutils
- Logování příkazů a jejich efektů
- rm implementováno pomocí FreeDestkop.org koše (trash-cli)

Ukázka: Comma-shell safe 1/2

```
n@t:/tmp/rm$ mkdir a
n@t:/tmp/rm$ touch a/b.c {d,e}.f g
n@t:/tmp/rm$ ,rm -fr *
,rm: Files to remove:
,rm:    Directory: "a" (with 1 files)
,rm:    Removing file: "/tmp/rm/g"
,rm:    2 files with extension "f"
,rm: Choose:
,rm:    [r]emove files
,rm:    [q]uit
,rm:    [t]rash files
,rm:    [s]how more files
t
,rm trash: trying to trash: "/tmp/rm/a" ok
,rm trash: trying to trash: "/tmp/rm/d.f" ok
,rm trash: trying to trash: "/tmp/rm/e.f" ok
,rm trash: trying to trash: "/tmp/rm/g" ok
,rm trash: Done. Use ,t or ,trash handle trashed bundles
```

Ukázka: Comma-shell safe 2/2

```
n@t:/tmp/rm$ ,t
,rm: This is a wrapper. Real trash is located at ~/.local/share/Trash
,rm: Choose the bundle:
    [1] ",rm a" from: /tmp/rm at 2017.05.03 16:53:39
    [2] ",rm -fr *" from: /tmp/rm at 2017.06.11 14:46:52
    [q]uit
,rm: Chosen bundle 2017-06-11-14-46-52-039333656.
    [a]nother - select different bundle
    [s]how all trashed files
    [r]estore
    [d]iscard from the trash
    [q]uit
Showing all deleted files:
    drwxrwxr-x 2 n n 4096 čen 11 14:46 /tmp/rm/a
    -rw-rw-r-- 1 n n 0 čen 11 14:46 /tmp/rm/d.f
    -rw-rw-r-- 1 n n 0 čen 11 14:46 /tmp/rm/e.f
    -rw-rw-r-- 1 n n 0 čen 11 14:46 /tmp/rm/g
```

Děkuji za pozornost

Shrnutí

- Hooks
- Debugger
- Safe příkazy

Otázky: oponent (1/2)

- Comma-shell používá pro vlastní spuštění trasovaných a debugovaných příkazů příkaz eval. Jak se vypořádá s použitím tohoto příkazu?

- Příkaz eval je možné spouštět rekurzivně.
- Comma-shell příkazem eval spustí více příkazů, ale na rekurzivní spouštění to nemá vliv.

Otázky: oponent (2/2)

- V práci jsem nenašel uživatelskou příručku s výčtem debugovatelných konstrukcí shellu. Můžete doplnit, které konstrukce Comma-shell podporuje a které ne?

- Konstrukce, se kterými umí Comma-shell debugger (v interaktivním shellu) pracovat, jsou: roura, subshell, cykly for a while
- Přidání dalších konstrukcí, se kterými umí pracovat knihovna bashlex, je snadné