Fuzz testování překladačů

Petr Muller

7. listopadu 2008

Pozadí práce

Práce

- Bakalářská práce
- Cílem bylo zjistit, zda je možné fuzz testování aplikovat na překladače
- Byl implementován nástroj využívající tuto metodu

Metoda fuzz testování

- Jednoduchá automatizovaná metoda pro testování robustnosti aplikací
- Princip:
  - Jako vstup programu se použije (pseudo) náhodný řetezec znaků
  - Sleduje se, zda program zvládne takový stav zpracovat
  - Pokud program zareaguje, testem prošel. Pokud ne (pád, zacyklení), pak neprošel
- Metoda zvládá rychle odhalit nedostatečné ošetření vstupu
- Užitečné při testování, jak program nakládá s nedůvěryhodnými (vzdálenými) vstupy
- Nedostatečné ošetření vstupu je hlavním zdrojem bezpečnostních problémů software

Typy fuzz testování

Čistý fuzz

- Použije se zcela náhodný řetězec
- Triviální implementace
- Testuje pouze p̈ovrchp̈rogramu

Fuzzing formátu

- Se používá v případě, že vstup je v nějakém jazyce nebo protokolu (HTML, prog. jazyk...)
- Vstupem je náhodná, avšak gramaticky správná věta jazyka vstupu
- Vstup projde "hlouběji" do testovaného programu

Fuzz testování překladačů

Čistý fuzz

- Nevhodné, testuje pouze lexikální analyzátor

Fuzz jazyka C

- Náhodné, avšak validní programy v jazyce C
- Testují odolnost velké části kódu překladače
- Je možné touto metodou nalézt chyby, které se projevují v čase překladu

Chyby v překladačích

Méně závažné

Nebrání správnému překladu (chybějící/přebývající varování, výkonnostní problémy)

Středně závažné

- Projevují se při překladu
- Brání překladu (odmítnutí platného kódu, ICE[a])

[a] Internal Compiler Error

Velmi závažné chyby

- Projevují se až ve špatném chování výsledného programu
- Obtížné odhalit pravou příčinu (přijetí neplatného kódu, generování špatného kódu)

Hledání nejzávažnějších chyb pomocí fuzz testování

- Není možné určit konkrétní správný výstup překladače
- Správnost výstupu překladače je dána správností výstupu výsledného programu
- Jak určit správnost výstupu náhodného programu?

Fuzz testování překladače s porovnáváním

Náhodný program musí mít deterministický výstup

1. Program je přeložen dvakrát: testovaným a referenčním překladačem
2. Oba přeložené programy se spustí a porovná se jejich výstup
3. Je nepravděpodobné, že se stejná chyba vyskytuje v obou překladačích
4. Pokud se výstupy liší, pak se v jednom z překladačů nachází chyba

fucc - Fuzzing C Compiler

Nástroj pro realizaci popsané metody

- Generátor náhodných vět
- Překlad oběma překladači
- Tvorba potřebných výstupů
- Porovnávání výstupů

Projekt

- Implementováno v Pythonu + shell scripty
- Open-source, repozitář http://git.afri.cz/git/fucc.git

Aktuální vlastnosti

Vlastnosti

- Generátor je schopen tvořit platné, ne zbytečně složité programy
- Některé části lze konfigurovat
- Hledá rozdíly ve výsledku překladu, výstupu a ukončení programu
- Je schopen rozeznat některé obvyklé chyby generátoru

Výsledky

- Program byl testován pouze jednou, nalezeno bylo cca 10 chyb v TCC, a dvou verzích GCC

Budoucí vývoj

Odstranění chyb

- Odstranit některé případy vygenerovaného neplatného kódu
- Lépe integrovat jednotlivé součásti
- Zredukovat jazykově závislou část generátoru

Vlastnosti

- Lepší konfigurovatelnost porovnávání
- Usnadnění ověřování chyb
- Podpora regresního testování
- Grafické uživatelské rozhraní

Závěr

Závěr

Otázky?

Děkuji za pozornost!