Aktualizácia kódu procesu pomocou GDB

Filip Koseček

Školiteľ: Ing. Dušan Bernát, PhD.

Úvod

- aplikovanie záplat na bežiaci proces
- závislé od cieľovej architektúry procesora a operačného systému
- x86-64, GNU/Linux
- nahrádzanie celých funkcií

Motivácia

- minimalizácia času nedostupnosti servera
- pri dlhých výpočtoch môže byť časovo drahé spustiť proces nanovo

Výsledky

- rozšírenie nástroja GDB nasledovnými funkciami:
  - nahrádzanie celých funkcií (definovaných používateľom alebo z dynamicky linkovaných knižníc)
  - ukladanie metadát o záplatách v pamäti cieľového procesu
  - tabuľka záznamov o vykonaných záplatách (história vykonaných záplat)
  - podpora opakovaného aplikovania záplat (funkcie môžu byť nahradené viacerými záplatami)
  - opätovná aktivácia neaktívnej záplaty
  - odstránenie záplaty - vrátenie funkcie (a potenciálne celého procesu) do pôvodného stavu

Výsledky

- rozšírenie nástroja GDB nasledovnými funkciami:
  - nahrádzanie celých funkcií (definovaných používateľom alebo z dynamicky linkovaných knižníc)
  - ukladanie metadát o záplatách v pamäti cieľového procesu
  - tabuľka záznamov o vykonaných záplatách (história vykonaných záplat)
  - podpora opakovaného aplikovania záplat (funkcie môžu byť nahradené viacerými záplatami)
  - opätovná aktivácia neaktívnej záplaty
  - odstránenie záplaty - vrátenie funkcie (a potenciálne celého procesu) do pôvodného stavu

Výsledky

- rozšírenie nástroja GDB nasledovnými funkciami:
  - nahrádzanie celých funkcií (definovaných používateľom alebo z dynamicky linkovaných knižníc)
  - ukladanie metadát o záplatách v pamäti cieľového procesu
  - tabuľka záznamov o vykonaných záplatách (história vykonaných záplat)
  - podpora opakovaného aplikovania záplat (funkcie môžu byť nahradené viacerými záplatami)
  - opätovná aktivácia neaktívnej záplaty
  - odstránenie záplaty - vrátenie funkcie (a potenciálne celého procesu) do pôvodného stavu

Výsledky

- rozšírenie nástroja GDB nasledovnými funkciami:
  - nahrádzanie celých funkcií (definovaných používateľom alebo z dynamicky linkovaných knižníc)
  - ukladanie metadát o záplatách v pamäti cieľového procesu
  - tabuľka záznamov o vykonaných záplatách (história vykonaných záplat)
  - podpora opakovaného aplikovania záplat (funkcie môžu byť nahradené viacerými záplatami)
  - opätovná aktivácia neaktívnej záplaty
  - odstránenie záplaty - vrátenie funkcie (a potenciálne celého procesu) do pôvodného stavu

Výsledky

- rozšírenie nástroja GDB nasledovnými funkciami:
  - nahrádzanie celých funkcií (definovaných používateľom alebo z dynamicky linkovaných knižníc)
  - ukladanie metadát o záplatách v pamäti cieľového procesu
  - tabuľka záznamov o vykonaných záplatách (história vykonaných záplat)
  - podpora opakovaného aplikovania záplat (funkcie môžu byť nahradené viacerými záplatami)
  - opätovná aktivácia neaktívnej záplaty
  - odstránenie záplaty - vrátenie funkcie (a potenciálne celého procesu) do pôvodného stavu

Výsledky

- rozšírenie nástroja GDB nasledovnými funkciami:
  - nahrádzanie celých funkcií (definovaných používateľom alebo z dynamicky linkovaných knižníc)
  - ukladanie metadát o záplatách v pamäti cieľového procesu
  - tabuľka záznamov o vykonaných záplatách (história vykonaných záplat)
  - podpora opakovaného aplikovania záplat (funkcie môžu byť nahradené viacerými záplatami)
  - opätovná aktivácia neaktívnej záplaty
  - odstránenie záplaty - vrátenie funkcie (a potenciálne celého procesu) do pôvodného stavu

Implementácia

- GDB python script
- pridané príkazy:
  - `patch` - vykonanie záplaty
  - `patch-log` - vypísanie histórie vykonaných záplat
  - `patch-dump` - uloženie histórie vykonaných záplat do súboru
  - `patch-reapply` - opätovná aktivácia a odstánenie záplat

Záplaty

- používateľ deklaruje a implementuje funkcie, ktoré nahradia funkcie v cieľovom procese
- definuje dvojicu - pôvodná a náhradná funkcia
- zdrojový kód záplaty musí byť skompilovaný ako dynamicky linkovaná knižnica
- nahranie do pamäte pomocou `dlopen`

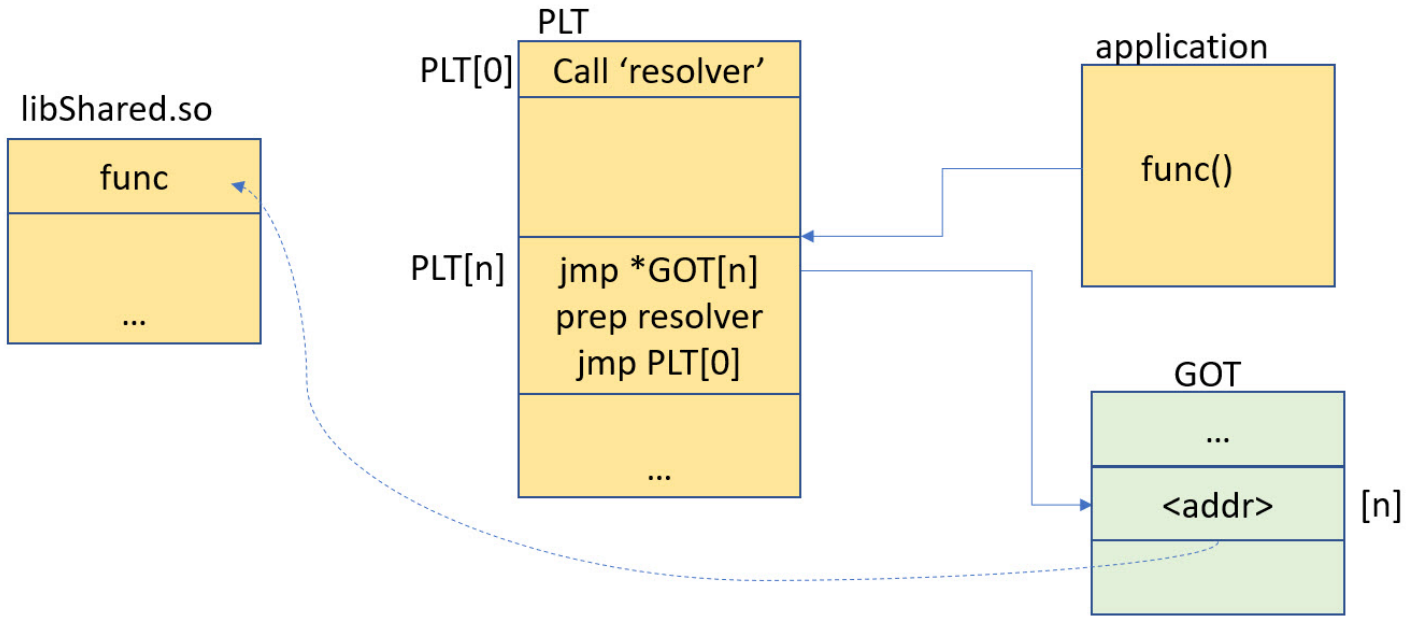
Nahrádzanie funkcií definovaných používateľom

- trampolíny - mechanizmus na presmerovanie riadenia toku do náhradnej funkcie
- príklad trampolíny:

```
movabs $patch_func_ptr, %r11
jmp *%r11
```

Nahrádzanie funkcií z dynamicky linkovaných knižníc

- prepísanie položky v GOT tabuľke

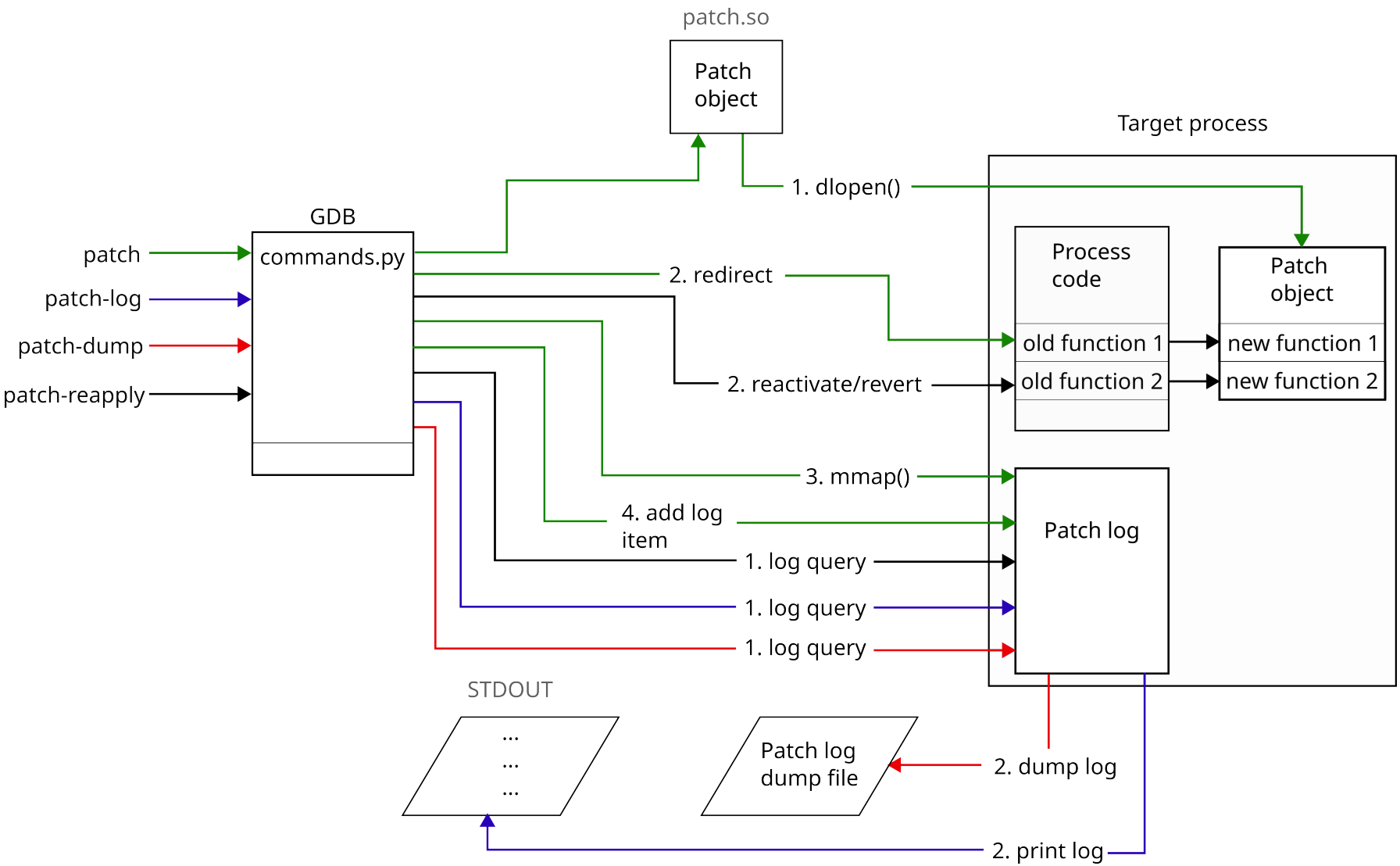


Linux lazy binding

Ukladanie metadát pre opakované vykonávanie záplat

- perzistencia metadát
- hlavička v špeciálnej sekcii
- alokovanie samostatných pamäťových stránok v cieľovom procese pre samotné metadáta
- "master library"

Implementácia



Zhrnutie

- preskúmanie možností ako vykonávať modifikáciu kódu procesu pomocou GDB
- návrh a implementácia nástroja, ktorý umožňuje vykonávať spomenutú funkcionalitu bez akejkoľvek podpory cieľového procesu
- možné rozšírenia

Absolútna trampolína

```
#AT&T
movabs $patch_function_ptr, %r11
jmp *%r11

#Intel
movabs r11, patch_function_ptr
jmp r11

#machine code
0:  49 bb 00 00 00 00 00    movabs r11,0x0
7:  00 00 00
a:  41 ff e3                jmp    r11
```

Zatvorená knižnica

```
(gdb) patch dec.so --log
(gdb) patch-log
[0] revert
[1]* 2023-06-19 09:00:18: target_function -> dec.so:patch_function
(gdb) patch mult.so --log
(gdb) patch-log
[0] revert
[1] 2023-06-19 09:00:18: target_function -> dec.so:unknown function
(library is closed)
[2]* 2023-06-19 09:00:37: target_function -> mult.so:patch_function
(gdb) patch-reapply 1
The library has been closed. Cannot apply the patch.
To apply the patch, use patch command.
```