

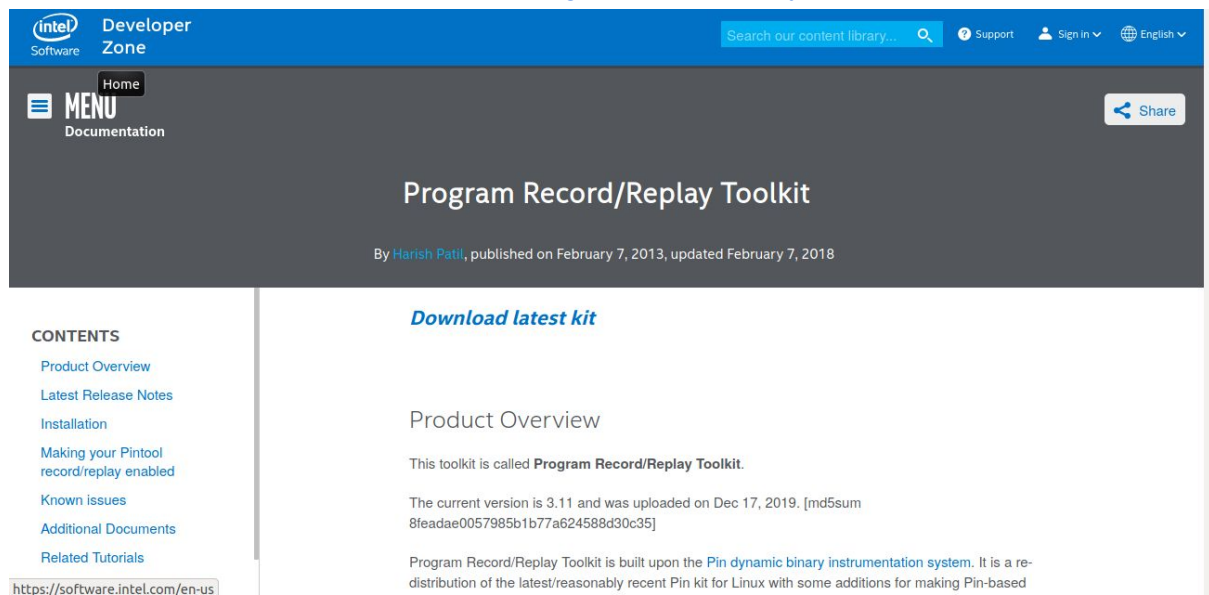
SINUCA-tracer

Este documento descreve os passos necessários para a utilização do SINUCA-tracer dentro do OrCS para gerar traços de execução de aplicações ou executar qualquer aplicação diretamente com o OrCS.

Download do Pin + PinPlay

A versão mais recente dessas ferramentas pode ser baixada no seguinte endereço:


<https://software.intel.com/en-us/articles/program-recordreplay-toolkit>



The screenshot shows the Intel Developer Zone website. The header is blue with the Intel logo, 'Developer Zone' text, a search bar, and links for Support, Sign in, and English. Below the header is a dark grey banner with the title 'Program Record/Replay Toolkit' and a byline 'By Harish Patil, published on February 7, 2013, updated February 7, 2018'. A 'Share' button is on the right. On the left, a 'MENU' sidebar lists: Home, Product Overview, Latest Release Notes, Installation, Making your Pintool record/replay enabled, Known issues, Additional Documents, and Related Tutorials. The main content area has a 'Download latest kit' link and a 'Product Overview' section. The overview text states: 'This toolkit is called Program Record/Replay Toolkit. The current version is 3.11 and was uploaded on Dec 17, 2019. [md5sum 8feadae0057985b1b77a624588d30c35] Program Record/Replay Toolkit is built upon the Pin dynamic binary instrumentation system. It is a re-distribution of the latest/reasonably recent Pin kit for Linux with some additions for making Pin-based

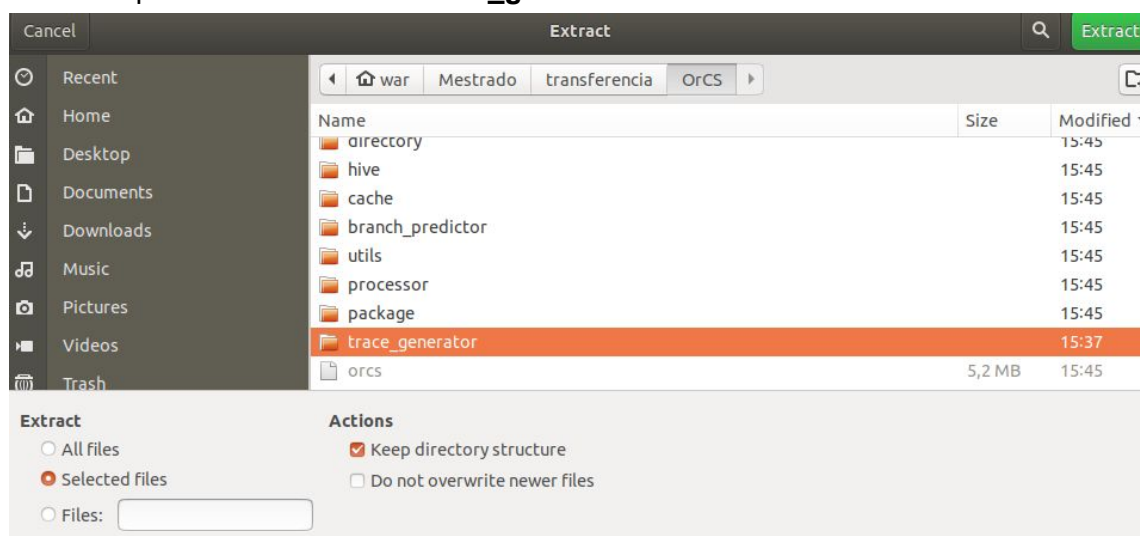
Extração de arquivos

Os arquivos presentes no arquivo compactado baixado anteriormente devem ser selecionados:



Name	Size	Type	Modified
doc	6,6 MB	Folder	22 julho 2019, 20:56
extras	78,0 MB	Folder	17 dezembro 2019, 10:03
ia32	39,1 MB	Folder	22 julho 2019, 20:54
intel64	45,8 MB	Folder	22 julho 2019, 20:54
licensing	661,3 kB	Folder	22 julho 2019, 20:55
source	21,9 MB	Folder	22 julho 2019, 21:08
pin	245,3 kB	unknown	22 julho 2019, 21:08
pin.sig	7,8 kB	detached O...	22 julho 2019, 21:08
README	50,1 kB	unknown	22 julho 2019, 20:49

E descompactados no diretório **trace_generator** do OrCS:



Ajustes finais

Dentro do diretório **trace_generator** os seguintes comandos devem ser executados:

```
>> chmod +x ./install.sh  
>> ./install.sh
```

Esses comandos vão tornar o arquivo **install.sh** executável e executar esse arquivo para realizar as configurações finais do gerador de traços.

Compilando

Para compilar o gerador de traços, navegue até a pasta:

```
>> trace_generator/extras/pinplay/sinuca_tracer
```

E digite os seguintes comandos:

```
>> make clean
```

```
>> make
```

O resultado esperado da compilação sem erros se parecerá com isto:

```
*****
Moving instruction_type_count.so to /home/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/bin/intel64/
mv obj-intel64/instruction_type_count.so /home/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/bin/intel64/

g++ -shared -Wl,-hash-style=sysv /home/war/Mestrado/transferencia/OrCS/trace_generator/intel64/runtime/pincrt/crtbeginS.o -Wl,-Bsymbolic -Wl,
,-version-script=/home/war/Mestrado/transferencia/OrCS/trace_generator/source/include/pin/pintool.ver -fabi-version=2 -o obj-intel64/lib
rary_call.so obj-intel64/library_call.o /home/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/lib/intel64/libpinplay.a /home
/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/lib-ext/intel64/libbz2.a /home/war/Mestrado/transferencia/OrCS/trace_generat
or/extras/pinplay/lib-ext/intel64/libzlib.a /home/war/Mestrado/transferencia/OrCS/trace_generator/source/tools/InstLib/obj-intel64/controll
er.a -L/home/war/Mestrado/transferencia/OrCS/trace_generator/intel64/runtime/pincrt -L/home/war/Mestrado/transferencia/OrCS/trace_generator
/intel64/lib -L/home/war/Mestrado/transferencia/OrCS/trace_generator/intel64/lib-ext -L/home/war/Mestrado/transferencia/OrCS/trace_generator
/extras/xed-intel64/lib -lpin -lxed /home/war/Mestrado/transferencia/OrCS/trace_generator/intel64/runtime/pincrt/crtendS.o -lpin3dwarf -ldl-
dynamic -nostdlib -lstdport-dynamic -ln-dynamic -lc-dynamic -lunwind-dynamic

*****
Moving library_call.so to /home/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/bin/intel64/
mv obj-intel64/library_call.so /home/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/bin/intel64/

g++ -shared -Wl,-hash-style=sysv /home/war/Mestrado/transferencia/OrCS/trace_generator/intel64/runtime/pincrt/crtbeginS.o -Wl,-Bsymbolic -Wl
,-version-script=/home/war/Mestrado/transferencia/OrCS/trace_generator/source/include/pin/pintool.ver -fabi-version=2 -o obj-intel64/pro
file.so obj-intel64/profile.o /home/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/lib/intel64/libpinplay.a /home/war/Mestr
ado/transferencia/OrCS/trace_generator/extras/pinplay/lib-ext/intel64/libbz2.a /home/war/Mestrado/transferencia/OrCS/trace_generator/extras
/pinplay/lib-ext/intel64/libzlib.a /home/war/Mestrado/transferencia/OrCS/trace_generator/source/tools/InstLib/obj-intel64/controller.a -L/h
ome/war/Mestrado/transferencia/OrCS/trace_generator/intel64/runtime/pincrt -L/home/war/Mestrado/transferencia/OrCS/trace_generator/intel64/l
ib -L/home/war/Mestrado/transferencia/OrCS/trace_generator/intel64/lib-ext -L/home/war/Mestrado/transferencia/OrCS/trace_generator/extras/x
ed-intel64/lib -lpin -lxed /home/war/Mestrado/transferencia/OrCS/trace_generator/intel64/runtime/pincrt/crtendS.o -lpin3dwarf -ldl-dynamic -n
ostdlib -lstdport-dynamic -ln-dynamic -lc-dynamic -lunwind-dynamic

*****
Moving profile.so to /home/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/bin/intel64/
mv obj-intel64/profile.so /home/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/bin/intel64/

make[1]: Leaving directory '/home/war/Mestrado/transferencia/OrCS/trace_generator/extras/pinplay/sinuca_tracer'
```

Criando traços

Após a compilação, traços podem ser gerados através do seguinte comando, executado a partir da pasta **trace_generator/extras/pinplay/sinuca_tracer**:

```
>> ../../pin -t ../bin/intel64/sinuca_tracer.so -trace x86 -- <<Caminho até o executável>>
```

Exemplo:

```
>> ../../pin -t ../bin/intel64/sinuca_tracer.so -trace x86 -- /bin/ls
```

Esse comando gera uma saída parecida com a seguinte:

```
gcc threads = 1
Inserted Output File Name = output_trace.out
Real Static File = output_trace.out.tid0.stat.out.gz => READY !
Real Dynamic File = output_trace.out.tid0.dyn.out.gz => READY !
Real Memory File = output_trace.out.tid0.mem.out.gz => READY !
Loading /bin/ls, Image id = 1
Loading /lib64/ld-linux-x86-64.so.2, Image id = 2
Loading [vdso], Image id = 3
Loading /lib/x86_64-linux-gnu/libselinux.so.1, Image id = 4
Loading /lib/x86_64-linux-gnu/libc.so.6, Image id = 5
Loading /lib/x86_64-linux-gnu/libpcre.so.3, Image id = 6
Loading /lib/x86_64-linux-gnu/libdl.so.2, Image id = 7
Loading /lib/x86_64-linux-gnu/libpthread.so.0, Image id = 8
branch_profiler.cpp      intrinsics_extension.hpp  obj-ia32                output_trace.out.tid0.mem.out.gz  profile.cpp
defines.hpp              library_call.cpp           obj-intel64             output_trace.out.tid0.stat.out.gz  sinuca_tracer.cpp
enumerations.hpp         makefile                   opcode_package.hpp       pinplay-debugger-shell.cpp
instruction_type_count.cpp makefile.rules              opcodes.hpp             pinplay-debugger-shell.H
intrinsic_extensions.cpp  memory_request_client.hpp  output_trace.out.tid0.dyn.out.gz pinplay-driver.cpp
```

E cria 3 arquivos .gz que são os traços criados que podem ser utilizados como entrada para simulações com a atual versão do OrCS.

Executando diretamente

Um programa pode ser executado diretamente sobre o OrCS. Para isso, o OrCS deve ser compilado com os comandos:

```
>> make clean
```

```
>> make
```

(O sinuca tracer também deve ter sido compilado)

Em seguida, o seguinte comando pode ser utilizado na pasta raiz do OrCS (a mesma do make que compilou o OrCS):

```
>> trace_generator/pin -t trace_generator/extras/pinplay/bin/intel64/direct_tracer.so -c  
<arquivo de configuração do OrCS> -f <resultados do OrCS> -- <programa a ser  
executado> <argumentos do programa>
```

Por exemplo, com o seguinte comando podemos simular o ls sendo executado em uma arquitetura Sandy Bridge com o OrCS:

```
>> trace_generator/pin -t trace_generator/extras/pinplay/bin/intel64/direct_tracer.so -c  
config/sandy_bridge/sandy_bridge.cfg -f result.temp -- /bin/l
```

Notas:

- Caso o arquivo de configuração não seja informado, a configuração padrão é a do Sandy Bridge.
- Apenas aplicações single thread podem ser executadas desta maneira.
- A saída padrão do programa simulado é o stdout (seu terminal).
- Caso o arquivo de resultados não seja definido (flag -f), o arquivo **results.res** será utilizado como argumento -f para o OrCS.

Versões utilizadas:

Pin + PinPlay	3.11
GCC	7.5.0