

Платформенно-независимый и масштабируемый инструмент поиска клонов кода в бинарных файлах программ

Институт системного программирования РАН

А.К. Асланян : hayk@ispras.ru

Ш.Ф. Курмангалеев : kursh@ispras.ru>

В.Г. Варданян : vaag@ispras.ru

М.С. Арутюнян : arutunian@ispras.ru

С.С. Саргсян : sevaksargsyan@ispras.ru

Типы клонов для бинарного кода

1. Фрагменты кода, которые полностью совпадают
2. Фрагменты кода, которые могут отличаться типами, значениями данных, именами регистров
3. Фрагменты кода, которые могут отличаться типами, значениями данных именами регистров, а также могут отличаться некоторыми инструкциями (в конкретном фрагменте могут присутствовать или отсутствовать некоторые инструкции)

Примеры клонов бинарного кода (соответствующий ассемблер)

Фрагмент кода	Клон типа 1	Клон типа 2	Клон типа 3
<pre>public main main proc near var_4= dword ptr -4 argc= dword ptr 8 argv= dword ptr 0Ch envp= dword ptr 10h push ebp mov ebp, esp mov [ebp+var_4], 5 mov eax,[ebp+var_4] imul eax,[ebp+var_4] leave retn main endp</pre>	<pre>public main main proc near var_4= dword ptr -4 argc= dword ptr 8 argv= dword ptr 0Ch envp= dword ptr 10h push ebp mov ebp, esp mov [ebp+var_4], 5 mov eax,[ebp+var_4] imul eax,[ebp+var_4] leave retn main endp</pre>	<pre>public main main proc near var_4= dword ptr -4 argc= dword ptr 8 argv= dword ptr 0Ch envp= dword ptr 10h push ebp mov ebp, esp mov [ebp+var_4], 10 mov ecx, [ebp+var_4] imul ecx, [ebp+var_4] leave retn main endp</pre>	<pre>public main main proc near var_1= dword ptr -4 argc= dword ptr 8 argv= dword ptr 0Ch envp= dword ptr 10h push ebp mov ebp, esp mov [ebp+var_1], 15 mov ecx, [ebp+var_1] leave retn main endp</pre>

Возможное применение технологии

Поиск:

1. Вредоносного кода
2. Семантических ошибок в коде
3. Нарушений условий лицензионных соглашений на использование кода библиотек

Существующие методы

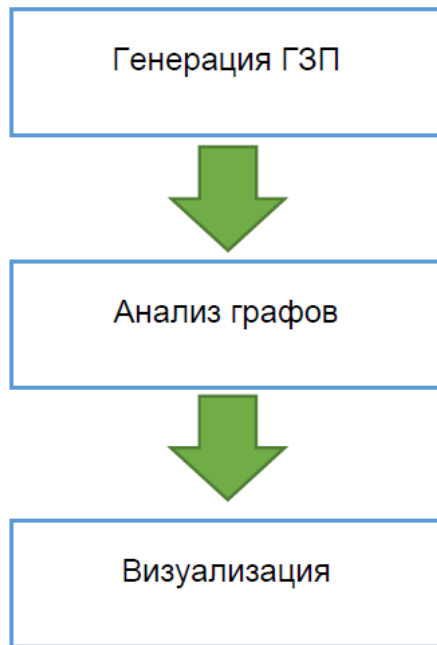
1. J. Jang and D. Brumley. Bitshred: Fast, scalable code reuse detection in binary code (cmu-cylab-10-006). CyLab, page 28, 2009; (1 тип)
2. A. Schulman. Finding binary clones with opstrings function digests: Part III. Dr. Dobb's Journal, 30(9):64, 2005; (1 тип)
3. D. Bruschi, L. Martignoni, and M. Monga. Code normalization for self-mutating malware. IEEE Security & Privacy, 5(2):46–54, 2007. (1,2 типы)
4. Thomas Dullien, Ero Carrera, Soeren-Meyer Eppler, Sebastian Porst «Automated attacker correlation for malicious code» // Technical report, DTIC Document, 2010 (1, 2 типы)
5. A. Sæbjørnsen, J. Willcock, T. Panas, D. Quinlan, and Z. Su. Detecting code clones in binary executables. In Proceedings of the 18th International Symposium on Software Testing and Analysis, pages 117–128. ACM, 2009. (1, 2, 3 типы)

Постановка задачи

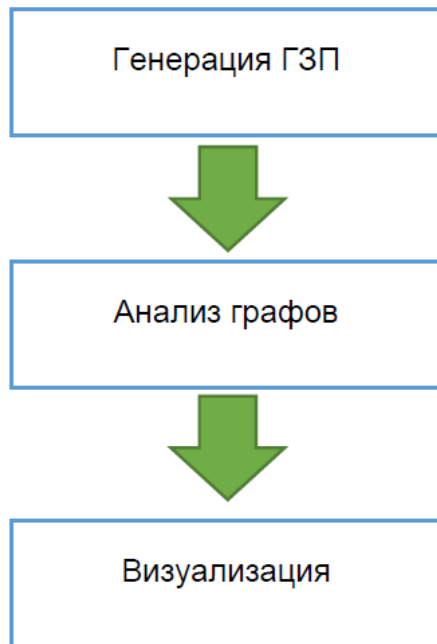
Разработать и реализовать инструмент поиска клонов в бинарном коде программ удовлетворяющий следующим требованиям:

- Поиск 1, 2, 3 типов клонов
- Процент истинных срабатываний (>90%)
- Независимость от целевой архитектуры (x86, x86-64, ARM, MIPS, PPC)
- Масштабируемый (размер бинарных файлов может достигать сотен МБ).

Архитектура инструмента



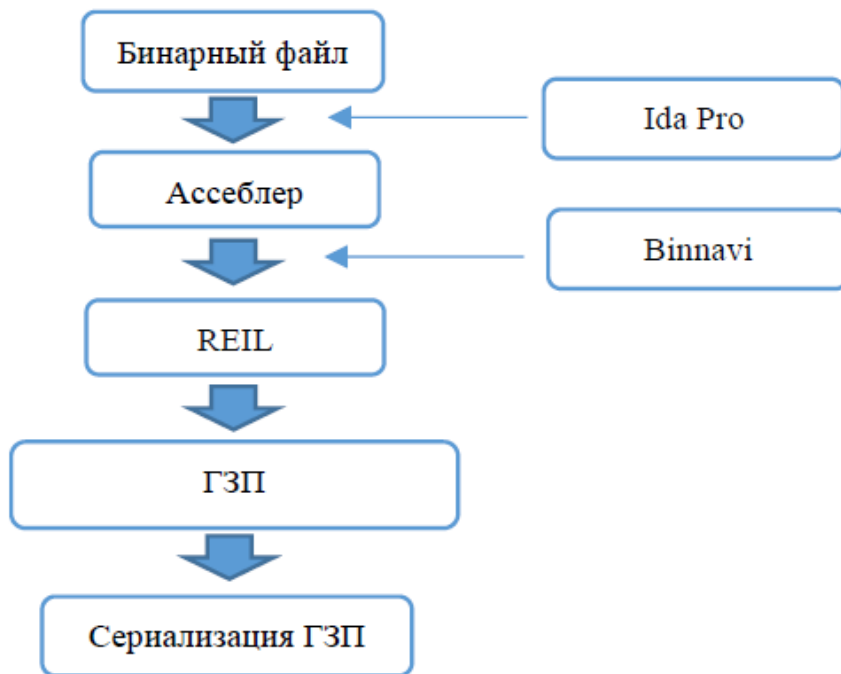
Архитектура инструмента



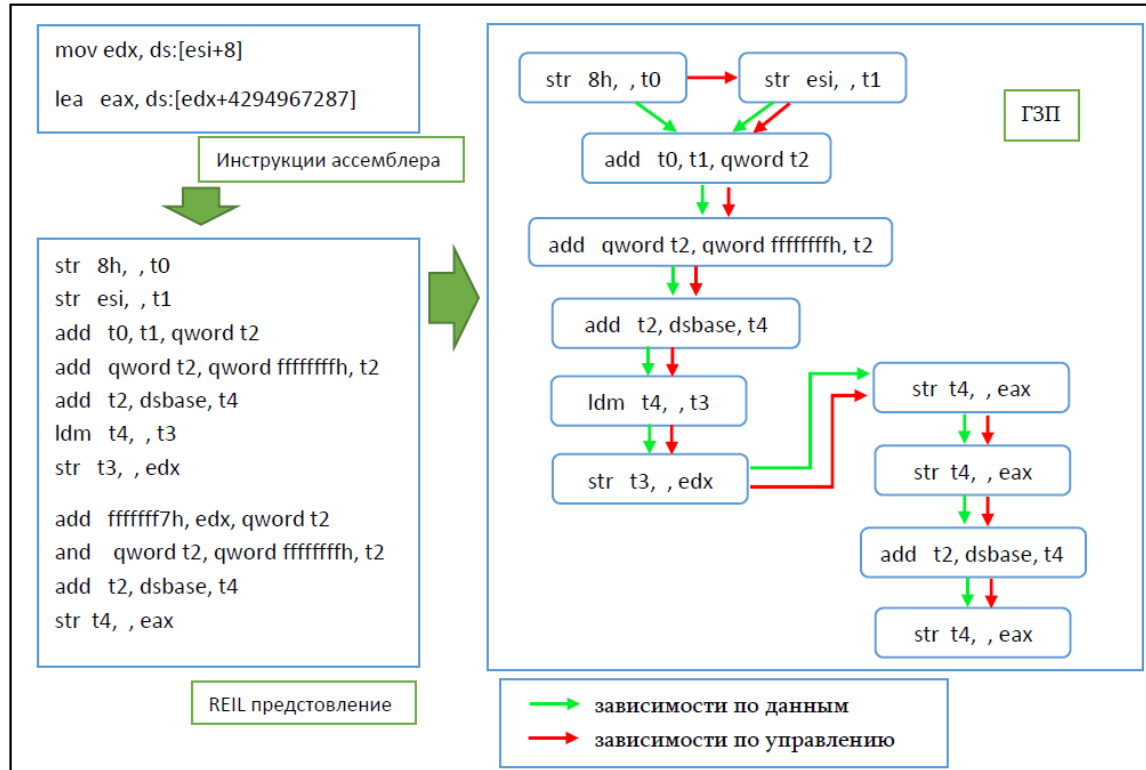
Технология обобщается для поиска клонов в исходном коде языков C/C++, JavaScript, Java.

1. С.С.Саргсян, диссертация, «Методы поиска клонов кода и семантических ошибок на основе семантического анализа программы»
2. Sargsyan S., Kurmangaleev S., Baloian A., Aslanyan H. Scalable and Accurate Clones Detection Based on Metrics for Dependence Graph // *Mathematical Problems of Computer Science*. 2014. Т. 42. С. 54-62
3. Саргсян С., Курмангалеев Ш., Белеванцев А., Асланян А., Балоян А. Масштабируемый инструмент поиска клонов кода на основе семантического анализа программ // *Труды Института системного программирования РАН*. 2015 Т. 27. № 1. С. 39-50.
4. Sargsyan S., Kurmangaleev S., Vardanyan V., Zakaryan V. Code Clones Detection Based on Semantic Analysis for JavaScript Language // *10th International Conference on Computer Science and Information Technologies*. 2015. С. 182-185.

Генерация ГЗП (1-ый этап)



Пример ГЗП



Анализ ГЗП (2-ой этап)

1. Загрузка ГЗП
2. Разделение ГЗП на подграфы (единицы сравнения)
3. Поиск наибольшего общего подграфа двух ЕС:
 - Фильтрация пар по параметру МЧИ
 - Сопоставление вершин и проверка удовлетворению МПС
4. Фильтрация результатов

Визуализация результатов (3-ий этап)

Code Clones Visualiser

File: lepton.idb

Choose Clones List

Save Clones

17. 29777-id-16-st-1475665332.clone

18. 29777-id-17-st-1475665332.clone

19. 29777-id-18-st-1475665332.clone

20. 29777-id-19-st-1475665332.clone

21. 29777-id-20-st-1475665332.clone

22. 29777-id-21-st-1475665332.clone

23. 29777-id-22-st-1475665332.clone

24. 29777-id-23-st-1475665332.clone

25. 29777-id-24-st-1475665332.clone

26. 29777-id-25-st-1475665332.clone

27. 29777-id-26-st-1475665332.clone

28. 29777-id-27-st-1475665332.clone

29. 29777-id-28-st-1475665332.clone

30. 29777-id-29-st-1475665332.clone

31. 29777-id-30-st-1475665332.clone

32. 29777-id-31-st-1475665332.clone

33. 29777-id-32-st-1475665332.clone

34. 29777-id-33-st-1475665332.clone

35. 29777-id-34-st-1475665332.clone

File: lepton.idb

```

/home/hayk/releaseCCD/tests/lepton.idb
ID : LeptonCodec::ThreadState::decode_row<ProbabilityTables<false,(BlockType)0>,Probability
(ProbabilityTables<false,(BlockType)0>)>,true>(ProbabilityTables<false,(BlockType)0> &,Prob
(ProbabilityTables<false,(BlockType)0>)> &,int,BlockBasedImagePerChannel<true> &,int)
*134646336: push_ebp
*134646337: push_edi
*134646338: push_esi
*134646339: push_ebx
*134646340: sub_esp_byte_44
*134646343: mov_eax_ss_esp_image_data
*134646347: mov_ebx_ss_esp_this
*134646351: mov_eax_ds_eax
*134646353: mov_eax_ds_eax_4
*134646356: test_eax_eax
*134646358: mov_ss_esp_var_34_eax
*134646362: jnz_134646376
*134646364: add_esp_byte_44
*134646367: pop_ebx
*134646368: pop_esi
*134646369: pop_edi
*134646370: pop_ebp
*134646371: retn
*134646376: mov_eax_ds_ebx_2164736
*134646382: mov_edx_ds_ebx_2164740
*134646388: lea_edi_ds_ebx_2164692
*134646394: mov_ecx_ds_ebx_2164744
*134646400: mov_esi_ds_ebx_2164748
*134646406: sub_esp_byte_4
*134646409: mov_ss_esp_decoder_edi
*134646413: mov_ss_esp_40h+context.above4+64_edx
*134646417: mov_ss_esp_40h+context.cur0+64_eax
*134646421: mov_ss_esp_40h+context.num_nonzeros_here8+64_ecx
*134646425: mov_ss_esp_40h+context.num_nonzeros_above12+64_esi
*134646429: push_ebx
*134646430: push_ss_esp_left_model
*134646434: push_edi
*134646435: push_esi
*134646436: push_ecx
*134646437: push_edx
*134646438: push_eax
*134646439: call_parse_tokens<false,(BlockType)0>(MBlockContext<AlignedBlock>,BoolDec
*134646444: mov_eax_ss_esp_image_data
*134646448: add_esp_byte_32
134646451: mov_ecx_ds_eax
134646453: mov_eax_ds_ebx_2164736

```

File: lepton.idb

```

/home/hayk/releaseCCD/tests/lepton.idb
ID : LeptonCodec::ThreadState::decode_row<ProbabilityTables<false,(BlockType)2>,Probability
(ProbabilityTables<false,(BlockType)2>)>,false>(ProbabilityTables<false,(BlockType)2> &,Prob
(ProbabilityTables<false,(BlockType)2>)> &,int,BlockBasedImagePerChannel<false> &,int)
*134653728: push_ebp
*134653729: push_edi
*134653730: push_esi
*134653731: push_ebx
*134653732: sub_esp_byte_44
*134653735: mov_eax_ss_esp_image_data
*134653739: mov_ebx_ss_esp_this
134653743: mov_eax_ds_eax_8
*134653746: mov_eax_ds_eax_4
*134653749: test_eax_eax
*134653751: mov_ss_esp_var_38_eax
*134653755: jnz_134653768
*134653757: add_esp_byte_44
*134653760: pop_ebx
*134653761: pop_esi
*134653762: pop_edi
*134653763: pop_ebp
*134653764: retn
*134653768: mov_eax_ds_ebx_2164768
*134653774: mov_edx_ds_ebx_2164772
*134653780: lea_edi_ds_ebx_2164692
*134653786: mov_ecx_ds_ebx_2164776
*134653792: mov_esi_ds_ebx_2164780
*134653798: sub_esp_byte_4
*134653801: mov_ss_esp_decoder_edi
*134653805: mov_ss_esp_40h+context.above4+64_edx
*134653809: mov_ss_esp_40h+context.cur0+64_eax
*134653813: mov_ss_esp_40h+context.num_nonzeros_here8+64_ecx
*134653817: mov_ss_esp_40h+context.num_nonzeros_above12+64_esi
*134653821: push_ebx
*134653822: push_ss_esp_left_model
*134653826: push_edi
*134653827: push_esi
*134653828: push_ecx
*134653829: push_edx
*134653830: push_eax
*134653831: call_parse_tokens<false,(BlockType)2>(MBlockContext<AlignedBlock>,BoolDec
*134653836: mov_eax_ss_esp_image_data
*134653840: mov_edi_ds_ebx_2164768
*134653846: add_esp_byte_32
134653849: mov_eax_ds_ebx_8

```

Результаты

Бинарный файл	Размер бинарного файла	Количество найденных клонов	Время работы инструмента
libxml2.so	1.8 МБ	2123	1м.46 с.
libfreetype.so	816 КБ	270	57 с.
openssl	764 КБ	40	35 с.
d8	33 МБ	40397	26 м.52 с.

Бинарные файлы получены компиляцией исходного кода с флагами -O0 -fno-inline, МЧИ равно 30, МПС равно 90%

Результаты

Бинарный файл	Размер бинарного файла	Количество найденных клонов	Время работы инструмента
libxml2.so	1.8 МБ	437	1м.25 с.
libfreetype.so	740 КБ	73	36 с.
openssl	672 КБ	71	26 с.
d8	20 МБ	19453	17 м.26 с.

Бинарные файлы получены компиляцией исходного кода с флагами -O3 -fno-inline, МЧИ равно 30, МПС равно 90%

Спасибо за внимание!