Analysis of Program Patches Nature and Searching for Unpatched Code Fragments

M. Arutunian, H. Aslanyan, V. Vardanyan, V. Sirunyan,
Sh. Kurmangaleev, S. Gaissaryan

SPLab, ISPRAS

Analysis of program patches

- Programs patches suppose fixing bugs or changing functionality
- Existing methods for analyzing patches (BinDiff, Diaphora, PatchDiff) require manual work
- Fragment with bugs can be propagated by copying and pasting it during program development

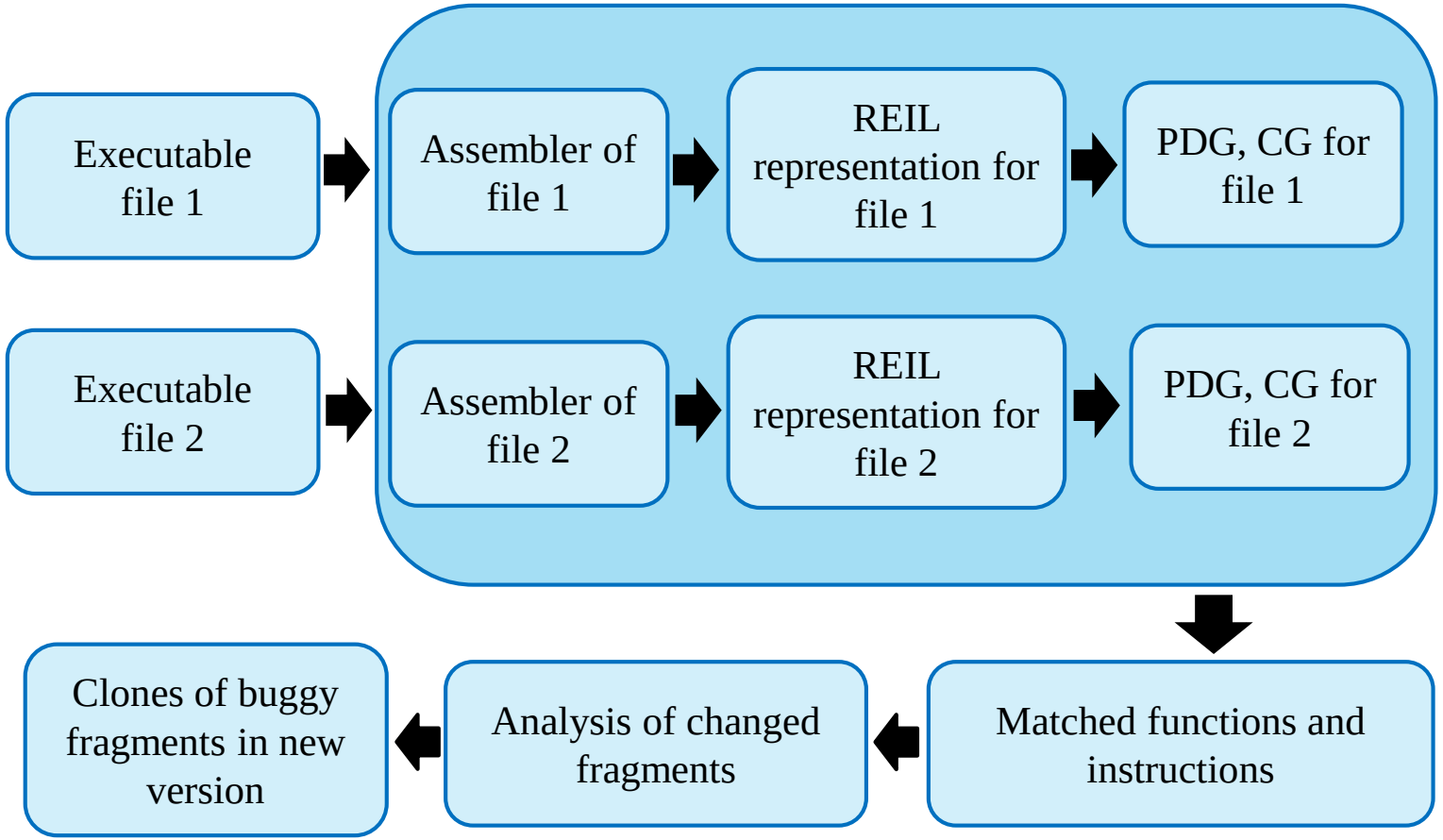
The purpose

- Develop a method for automatic analyzing the nature of patches between versions of executables
- Find unpatched code fragments using code clone detection

Related work

- SPAIN
- PVDF
- BinHunt
- iBinHunt

Structure of the tool



The Algorithm of executables comparison

Executables comparison tries to match functions and instructions from the first executable to functions and instructions from the second executable respectively.[1]

Executables comparison algorithm consists of two main steps:

- Match functions based on heuristics
- Match functions and instructions using algorithm for PDG maximum common subgraph detection

M. Arutunian, H. Aslanyan et al. "Scalable Framework for Accurate Binary Code Comparison" in *2017 Ivannikov ISPRAS Open Conference*, Moscow

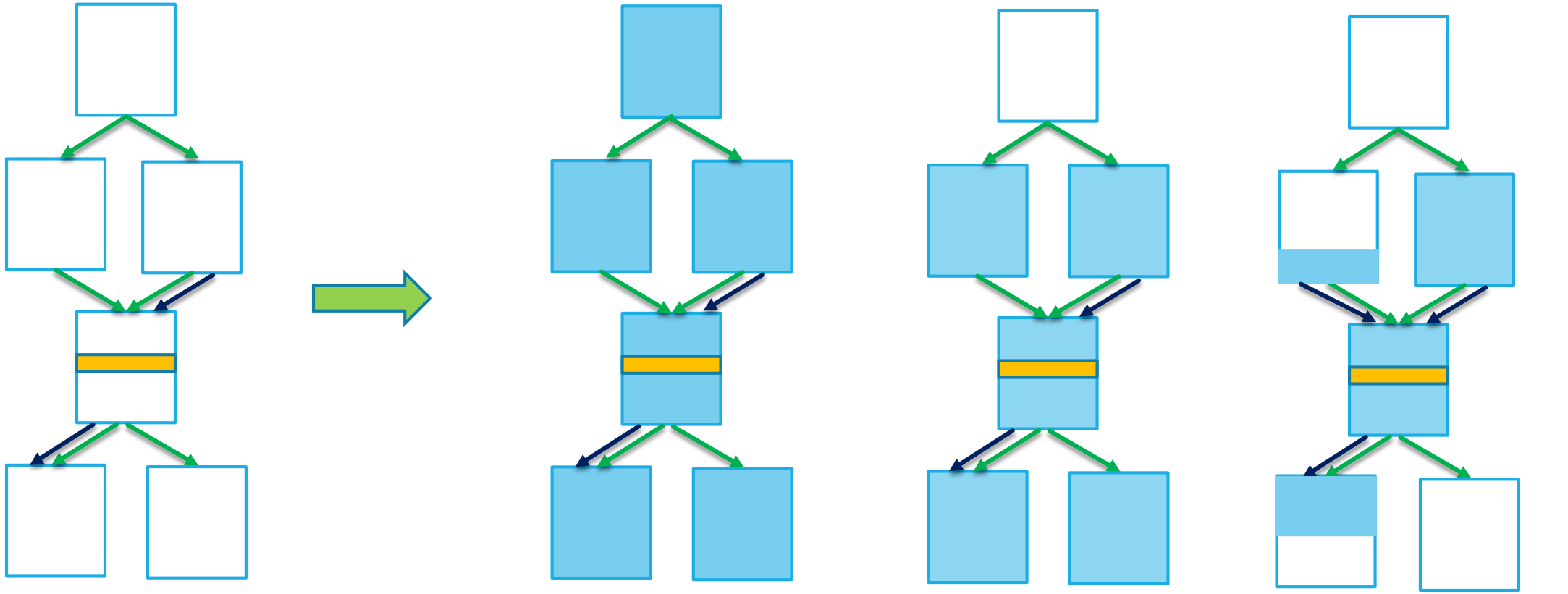
Analysis of the nature of changes in the new version of executable file

- Search for changed code fragment in the new version of the program
- Types of changes:
  - Function arguments are changed
  - Function call is changed
  - New basic block is added
  - New return instruction is added in a function
  - Break instruction is added in a loop
  - Continue instruction is added in a loop

Searching for unpatched defects

- Construction of the unpatched fragment in the old version
  - A fragment of an unpatched code is considered a changed function
  - A fragment of an unpatched code is considered a set of basic blocks
  - A fragment of an unpatched code is considered a set of instructions
- Search for clones of the unpatched fragment in the new version

Searching for unpatched defects



Function from old binary

Fragment is a whole function

Fragment is a set of basic blocks

Fragment is a set of instructions

Average percent of true positives

- DARPA cyber challenge test suit - 71.3%
- Corebench test suite - 73.3%.

Found patch example of DARPA's CGC_Planet_Markup_Language_Parser

```
                                        …
if ( end == -1 ) {
            cgc_printf ("!!Failed to locate … \n");
            return -1.0;  }
      temp = cgc_copyData( str, start, end );
#ifdef PATCHED
      if ( temp == NULL ) {
            return -1.0;  }
#endif
      if ( cgc_strcmp( temp, "Aphelion") != 0 ) {
            cgc_printf("!!Invalid cl… id: @s\n", temp);
            cgc_deallocate(temp, cgc_strlen(temp)+1);
            return aphelion; }
                                        …
```

Code fragment of cgc_extractAphelio function

Unpatched clone example of DARPA's CGC_Planet_Markup_Language_Parser

```
loc_804DB7F:
mov     eax, [ebp+str]
mov     ecx, [ebp+start]
mov     edx, [ebp+end]
mov     [esp], eax      ; str
mov     [esp+4], ecx    ; start
mov     [esp+8], edx    ; end
call    cgc_copyData
lea     ecx, aAphelion  ; "Aphelion"
mov     [ebp+temp], eax
mov     eax, [ebp+temp]
mov     [esp], eax      ; s1
mov     [esp+4], ecx    ; s2
call    cgc_strcmp
cmp     eax, 0
jz      loc_804DC12
```

```
loc_804DC6F:
mov     eax, [ebp+str]
mov     ecx, [ebp+start]
mov     edx, [ebp+end]
mov     [esp], eax      ; str
mov     [esp+4], ecx    ; start
mov     [esp+8], edx    ; end
call    cgc_copyData
mov     [ebp+temp], eax
cmp     [ebp+temp], 0
jnz     loc_804DCA7
```

```
loc_804DCA7:
lea     eax, aAphelion  ; "Aphelion"
mov     ecx, [ebp+temp]
mov     [esp], ecx      ; s1
mov     [esp+4], eax    ; s2
call    cgc_strcmp
cmp     eax, 0
jz      loc_804DD1E
```

Unpached and patched disassembly fragments of cgc_extractAphelio function

Unpatched clone example of DARPA's CGC_Planet_Markup_Language_Parser

```
                                        …
if ( end == -1 ) {
            cgc_printf ("!!Failed to locate …\n");
            return -1.0;  }
      temp = cgc_copyData( str, start, end );
#ifdef PATCHED
      if ( temp == NULL ) {
            return -1.0;  }
#endif
      if ( cgc_strcmp( temp, "Aphelion") != 0 ) {
            cgc_printf("!!Invalid cl… id: @s\n", temp);
            cgc_deallocate(temp, cgc_strlen(temp)+1);
            return aphelion; }
                                        …
```

Code fragment of cgc_extractAphelio function

```
                                        …
if ( end == -1 ) {
            cgc_printf ("!!Failed to locate …\n");
            return -1.0;  }
      temp = cgc_copyData( str, start, end );

      if ( cgc_strcmp( temp, “Radius") != 0 ) {
            cgc_printf("!!Invalid cl… id: @s\n", temp);
            cgc_deallocate(temp, cgc_strlen(temp)+1);
            return radius; }
                                        …
```

Code fragment of clone cgc_extractRadius function

Unpatched clone example of DARPA's CGC_Planet_Markup_Langugae_Parser

```
loc_804DB7F:
mov     eax, [ebp+str]
mov     ecx, [ebp+start]
mov     edx, [ebp+end]
mov     [esp], eax      ; str
mov     [esp+4], ecx    ; start
mov     [esp+8], edx    ; end
call    cgc_copyData
lea     ecx, aAphelion  ; "Aphelion"
mov     [ebp+temp], eax
mov     eax, [ebp+temp]
mov     [esp], eax      ; s1
mov     [esp+4], ecx    ; s2
call    cgc_strcmp
cmp     eax, 0
jz      loc_804DC12
```

Unpatched fragment of cgc_extractAphelio function from the ***unpatched version*** of executable

```
loc_804E993:
mov     eax, [ebp+str]
mov     ecx, [ebp+start]
mov     edx, [ebp+end]
mov     [esp], eax      ; str
mov     [esp+4], ecx    ; start
mov     [esp+8], edx    ; end
call    cgc_copyData
lea     ecx, aRadius    ; "Radius"
mov     [ebp+temp], eax
mov     eax, [ebp+temp]
mov     [esp], eax      ; s1
mov     [esp+4], ecx    ; s2
call    cgc_strcmp
cmp     eax, 0
jz      loc_804EA26
```

Cgc_extractRadius function's fragment from the ***patched version*** of the executable, which contains ***bug***

Results where unpatched fragment is detected

Project	Git commits		Function name with patched defect	Function name with unpatched defect
	Old version	New version		
Tcpdump	b534e304	d3aae719	juniper_monitor_print	1.juniper_mlfr_print
Tcpdump	c2ef6938	50a44b6b	ikev1_nonce_print	1.ikev1_hash_print 2.ikev1_sig_print 3.ikev1_ke_print 4.ikev1_vid_print
Libosip	79240bdd	a54f15b8	osip_www_ authenticate_init	1.sdp_connection_init 2.osip_authorization_init 3.osip_authentication _info_init
Libosip	80a955e7	03fe3a1c	osip_negotiation_sdp_ build_offer	1.osip_negotiation_sdp _build_offer

That's it

Thanks for your attention!