MobileFindr: Function Similarity Identification for Reversing Mobile Binaries

Yibin Liao, Ruoyan Cai, Guodong Zhu, Yue Yin, Kang Li



Reverse Engineering

• The process of taking a software program's binary code and recreating it so as to trace it back to the original source code

Reverse Engineering

• Purpose

- Find security flaws
- Test code compatibility
- Enhance new features without source
- Understand the design of malicious code

Tools for Reversing Mobile Apps



		📑 IDA View-A 🙁 🚺 Hex View-1 🗵 🖪	Structures 🗵 🗎 Enums 🗵 🛐 Imports 🗵 📝 Exports 🛿
Function	Eunction name	text: <mark>0137A1FA</mark> _var_11C	= -0x11C
		text: <mark>0137A1FA</mark> var_118	= -0x118
Window	f sub_13/4D30	text: <mark>0137A1FA</mark> var_114	- • • • • • • • • • • • • • • • • • • •
	f sub_13/4D46	text: <mark>0137A1FA</mark> var_110	= -0x110
	<u>f</u> sub_1374D5C	text: <mark>0137A1FA</mark> var_10B	= -0x10B
	<u>f</u> sub_1374D72	text: <mark>0137A1FA</mark> var_10A	= -0x10A
	f sub_1374D88	text: <mark>0137A1FA</mark> var_109	$= -0 \times 109$
	<u>f</u> sub_1374D9E	text: <mark>0137A1FA</mark> var_108	= -0x108 Accombly code
	f sub_1374DB4	text: <mark>0137A1FA</mark> var 8	= -8 ASSEILIDIY LOUE
	f sub_1374DCA	textij/A1FA	
	f sub_1374DE0	•*.xt: <mark>0137A1FA</mark>	PUSH {R4,R7,LR}
	f sub_1374DF6	•text:0137A1FC	ADD R7, SP, #4
	f sub_1374E0C	text:0137A1FE	SUB.W SP, SP, #0x228
	f sub 1374E22	text:0137A202	ADD R2, SP, #0x22C+var_14C
	f sub 1374E38	text:0137A204	MOUS R3, #0
	f sub 1374E4E	text:0137A206	MOV R9, #(stack_chk_guard_ptr - 0x137A212)
	f sub 1374F64	<pre>text:0137A20E</pre>	ADD R9, PC ;stack_chk_guard_ptr
	f sub 1374F7Δ	text:0137A210	LDR.W R9, [R9] ;stack_chk_guard
	f sub 137/F90	text:0137A214	LDR.W R9, [R9]
	f cub 1274EA9	text:0137A218	MOV R12, #(stack_chk_guard_ptr - 0x137A224)
	f sub_1277EE2	text:0137A220	ADD R12, PC ;stack_chk_guard_ptr
	7 SUD_1377FE2	text:0137A222	LDR.W R12, [R12] ;stack_chk_guard
	f sub_13/83/E	text:0137A226	LDR.W R12, [R12]
	f sub_13784B4	text:0137A22A	STR.W R12, [SP,#0x22C+var_8]
	f sub_13/8C/C	text:0137A22E	STR R3, [SP,#0x22C+var_14C]
	f sub_13/8F64	text:0137A230	STR R0, [SP,#0x22C+var_174]
	<u>f</u> sub_1378F90	text:0137A232	MOU RO, R2
	<u>f</u> sub_1379036	text:0137A234	LDR R2, [SP,#0x22C+var_174]
	fsub_137A1FA	text:0137A236	STR R1, [SP,#0x22C+var_178]
	<u>f</u> sub_137B4		MUV K1, K2
	f sub_137BA76		SIR.W Ry, [SP,#0x22C+Uar_17C]
	f sub_137D17C		BLX.W <u>J_ODJC_Storestrong_0</u>
	f sub_1380EB8		HUU K0, SP, #0X226+Vdr_150
	f sub_1380EC2	LEXL:0137H244	NUVS KI, #0
	f sub_1380ECC		SIN NI, [SF,#0X226*Vdr_150]
	f sub_1380ED6		DIX H i obje storeStrong B
	f sub_1380EE0	toyt • 01370240	MOULA RO #0x100 · cizo t
	f sub_1380EF6	text:01370252	$RIX \sqcup i mallor 0$
	f sub_1380F0C	text:01370256	MOU R1 #(objc msqSend ntr - 0x1370262)
	f sub 1380F22	• text:0137A25E	ADD R1 PC · objc_msgScha_pcr oxion2027
	f sub 138181A	• text:0137A260	LDR R1. [R1] : imp_objc_msqSend
	f sub 1381A4C	• text:01370262	MOU R2. $\#(selRef length - 0x137026E)$
	f sub 1381A6C	• text:0137A26A	ADD R2. PC : selRef length
	f sub 1381F2A	• text:0137A26C	STR R0, [SP,#0x22C+var 154]
	f sub 1381F68	text:0137A26E	LDR R0, [SP,#0x22C+var 1501
	f sub 1382080	text:0137A270	LDR R2, [R2] ; "lenath"
	f sub 1382782	text:0137A272	STR R1, [SP,#0x22C+var 180]
	f sub 1202P22	• text:0137A274	MOV R1, R2
	Sub_1302D22	• ext:0137A276	LDR R2, [SP,#0x22C+var 180]
	Line 167606 of 23790	0137A1FA 0157AIFA. Sub_157AIFA	

	📝 Func 🗖 🗗 🗙 []	IP were 🖾 🔽 htt View-1 🗵	\Lambda Structures 🗵	🗄 Enums 🛛 🕅 Imports 🖾 📝 Exports 🛙
	Function name	text: <mark>0137A1FA</mark> var_11C	= -0x11C	
	5 aut 1274D20	text: <mark>0137A1FA</mark> _var_118	= -0x118	
	7 SUD_1374D30	text: <mark>0137A1FA</mark> var_114	= -0x114	Disassembling Baidu Mobile (iOS)
	f sub_13/4040	text: <mark>0137A1FA</mark> va <mark>r_110</mark>	= -0x110	
	f sub_13/4D5C	text: <mark>0137A1FA</mark> va <mark>r</mark> _10B	$= -0 \times 10B$	
	f sub_13/4D/2	text: <mark>0137A1FA</mark> va <mark>r</mark> _10A	= -0x10A	
	f sub_1374D88	text: <mark>0137A1FA</mark> var_109	= -0x109	
	<u>f</u> sub_1374D9E	text: <mark>0137A1FA</mark> var_108	$= -0 \times 108$	
	f sub_1374DB4	text: <mark>0137A1FA</mark> va <mark>r</mark> _8	= -8	
	f sub_1374DCA	text: <mark>0137A1FA</mark>		
	f sub_1374DE0	text: <mark>0137A1FA</mark>	PUSH	{R4,R7,LR}
	f sub_1374DF6	text:0137A1FC	ADD	R7, SP, #4
	f sub_1374E0C	text:0137A1FE	SUB.W	SP, SP, #0x228
	f sub_1374E22	text:0137A202	ADD	R2, SP, #0x22C+var_14C
	f sub_1374E38	text:0137A204	MOUS	R3, #0
	f sub 1374E4E	text:0137A206	MOV	R9, #(stack_chk_guard_ptr - 0x137A212)
	f sub 1374E64	text:0137A20E	ADD	R9, PC ;stack_chk_guard_ptr
	f sub 1374E7A	text:0137A210	LDR.W	R9, [R9] ;stack_chk_guard
	f sub 1374F90	text:0137A214	LDR.W	R9, [R9]
	f sub 1374EA8	text:0137A218	MOV	R12, #(stack_chk_guard_ptr - 0x137A224)
	f sub 1277EE2	text:0137A220	ADD	R12, PC ;stack_chk_guard_ptr
	5 sub 127027E	text:0137A222	LDR.W	R12, [R12] ;stack_chk_guard
	J SUD_13/03/E	text:0137A226	LDR.W	R12, [R12]
	5 sub_1378484	text:0137A22A	STR.W	R12, [SP,#0x22C+var_8]
	f sub_1378C/C	text:0137A22E	STR	R3, [SP,#0x22C+var_14C]
	f sub_1378F04		STR	R0, [SP,#0x22C+var_174]
No dobug		text:013/A232	MUV	RU, R2
No debug	<u>f</u> sub_13/9036		LUK	K2, [SP,#0x220+Var_174]
symbols	f sub_137A1FA		218	K1, [SP,#0X220+Var_178]
Symbols	f sub 13784	LEXL:0137H238	PUV STD U	KI, KZ D0 _ CCD #00000.0000 4701
	<u>f</u> sub_137BA76	LEXL:0137H23H	51K.W	Ky, [Sr,#0x220+Vdr_170]
	f sub_137D17C	Lext.0137H23E	DLA.W	JUDJC_SCUPESCPUNG_0 DA_SD_#892204034 1EA
	f sub_1380EB8	• toyt • 01370244	MOUS	P1 #0
	<u>f</u> sub_1380EC2	• toyt • 01370244	CTR	D1 [CD #0v220+upv 150]
	f sub_1380ECC	• toyt • 01370240		R1 [SP #0x220.001_190]
	f sub_1380ED6	• toyt • 01370240	RIX M	i obje storeStrong Ø
	f sub_1380EE0	• text:0137A24F	MOLLA	R0 #0x100 · size t
	f sub_1380EF6	• text: 0137A252	BLX_W	i malloc Ø
	🗲 sub_1380F0C	• text: 01370256	MOU	R1, #(objc msgSend ntr - 0x1370262)
	f sub_1380F22	• text:0137A25E	ADD	R1. PC : obic msgSend otr
Need to	f sub_138181A	• text:0137A260	LDR	R1. [R1] : imp_objc_msgSend
	f sub 1381A4C	• text:0137A262	MOV	R2. #(selRef length - 0x137A26E)
determine	f sub 1381A6C	• text:0137A26A	ADD	R2, PC ; selRef length
determine	f sub 1381F2A	• text:0137A26C	STR	R0, [SP,#0x22C+var 154]
where to add	f sub 1381F68	• text:0137A26E	LDR	R0, [SP,#0x22C+var 150]
	f sub 1382080	text:0137A270	LDR	R2, [R2] ; "length"
breakpoints	f sub 1382782	<pre>text:0137A272</pre>	STR	R1, [SP,#0x22C+var_180]
	f sub 1382B22	<pre>text:0137A274</pre>	MOV	R1, R2
• IN 167,606		<pre>text:0137A276</pre>	LDR	R2, [SP,#0x22C+var_180]
functions				
Tunctions	Line 167606 of 23790	0137A1FA 0137A1FA: sub_137A1FA		

Baidu mobile (version 9.30) Function: sub_137A1FA

Baidu mobile (version 9.35) Function: sub_13290FA

f Func 🗖 🗗 🗙	📳 IDA View-A 🛛 🚺 Hex View-1	🗵 🖪 Structures 🗵	🔃 Enums 🛛 🕅 Imports 🖾 📝 Exports 🛽	📝 Functi 🗖 🗗 🗙	📘 IDA View-A 関	🖸 Hex View-1 🛛	\Lambda Structures 🗵 🔃	Enums 🗵 🎦 Imports 🗵 📝 Exports 🗵
Function name ^	text: <mark>0137A1FA</mark> var_11C	= -0x11C		Function name	_text	: <mark>:013290FA</mark> var_11C	= -0x11C	
1274D20	text: <mark>0137A1FA</mark> var_118	= -0x118			text	: <mark>013290FA</mark> var_118	= -0x118	
J SUD_1374D30	text: <mark>0137A1FA</mark> var_114	= -0x114		7 SUB_1323D90	_text	: <mark>:013290FA</mark> var_114	= -0x114	
5 SUD_1374D40	text: <mark>0137A1FA</mark> var_110	= -0x110		f sub_1323DA8	text	: <mark>:013290FA</mark> var_110	$= -0 \times 110$	
f sub_13/4D5C	text: <mark>0137A1FA</mark> var_10B	= -0x10B		f sub_1323E2	text	: <mark>:013290FA</mark> var_10B	$= -0 \times 10B$	
f sub_13/4D/2	text: <mark>0137A1FA</mark> var_10A	= -0x10A		<u>f</u> sub_13240E	text	: <mark>:013290FA</mark> var_10A	= -0x10A	
<u>f</u> sub_1374D88	text: <mark>0137A1FA</mark> var_109	$= -0 \times 109$		<u>f</u> sub_132418	text	: <mark>:013290FA</mark> var_109	$= -0 \times 109$	
<u>f</u> sub_1374D9E	text: <mark>0137A1FA</mark> var_108	$= -0 \times 108$		<u>f</u> sub_13242	text	: <mark>:013290FA</mark> var_108	$= -0 \times 108$	
<u>f</u> sub_1374DB4	text: <mark>0137A1FA</mark> var_8	= -8		<u>f</u> sub_132560	text	: <mark>:013290FA</mark> var_8	= -8	
f sub_1374DCA	text: <mark>0137A1FA</mark>			f sub_1325AC	text	:: <mark>013290FA</mark>		
f sub_1374DE0	text: <mark>0137A1FA</mark>	PUSH	{R4,R7,LR}	f sub_1325B6	text	: <mark>013290FA</mark>	PUSH	{R4,R7,LR}
f sub_1374DF6	text:0137A1FC	ADD	R7, SP, #4	f sub_132660	text	::013290FC	ADD	R7, SP, #4
f sub_1374E0C	text:0137A1FE	SUB.W	SP, SP, #0x228	f sub_1326AA	•text	:013290FE	SUB.W	SP, SP, #0x228
f sub_1374E22	text:0137A202	ADD	R2, SP, #0x22C+var_14C	f sub_1326B4	text	:01329102	ADD	R2, SP, #0x22C+var_14C
f sub_1374E38	text:0137A204	MOUS	R3, #0	f sub 1326BC	text	:01329104	MOUS	R3, #0
f sub_1374E4E	text:0137A206	MOV	R9, #(stack_chk_guard_ptr - 0x137A212)	f sub 1326C6	text	:01329106	MOV	R9, #(stack_chk_guard_ptr - 0x1329112)
f sub 1374E64	text:0137A20E	ADD	R9, PC ;stack_chk_guard_ptr	f sub 1326D0	text	::0132910E	ADD	R9, PC ;stack_chk_guard_ptr
7 sub 1374E7A	text:0137A210	LDR.W	R9, [R9] ;stack_chk_guard	f sub 1326EE2	text	:01329110	LDR.W	R9, [R9] ;stack_chk_guard
f sub 1374E90	text:0137A214	LDR.W	R9, [R9]	f sub 132727E	text	:01329114	LDR.W	R9, [R9]
f sub 1374FA8	text:0137A218	MOV	R12, #(stack_chk_guard_ptr - 0x137A224)	f cub 12272P4	text	:01329118	MOV	R12, #(stack_chk_guard_ptr - 0x1329124)
f sub 1377EE2	text:0137A220	ADD	R12, PC ;stack_chk_guard_ptr	5 sub_1327364	text	:01329120	ADD	R12, PC ;stack_chk_guard_ptr
f cub 127027E	text:0137A222	LDR.W	R12, [R12] ;stack_chk_guard	7 SUD_132767C	text	::01329122	LDR.W	R12, [R12] ;stack_chk_guard
J SUD_137037E	text:0137A226	LDR.W	R12, [R12]	f sub_1327E04	text	:01329126	LDR.W	R12, [R12]
5 SUD_1370464	text:0137A22A	STR.W	R12, [SP,#0x22C+var_8]	f sub_1327E90	text	::0132912A	STR.W	R12, [SP,#0x22C+var_8]
f sub_13/8C/C	text:0137A22E	STR	R3, [SP,#0x22C+var_14C]	<u>f</u> sub_1327F36	text	::0132912E	STR	R3, [SP,#0x22C+var_14C]
f sub_13/8F64	text:0137A230	STR	R0, [SP,#0x22C+var_174]	f_sub_13290FA	text	:01329130	STR	R0, [SP,#0x22C+var_174]
f sub_1378F90	text:013/A232	MUV	R0, R2	<u>f</u> sub_132A976	text	::01329132	MOV	R0, R2
<u>f</u> sub_1379036	text:013/A234	LDR	R2, [SP,#0x22C+var_174]	<u>f</u> sub_132C07C	text	::01329134	LDR	R2, [SP,#0x22C+var_174]
<u>f</u> _sub_137A1FA		STR	R1, [SP,#0x22C+var_178]	f sub_132FDB8	text	::01329136	STR	R1, [SP,#0x22C+var_178]
<u>f</u> sub_137B4		MUV	R1, K2	f sub_132FDC2	text	:01329138	MOV	R1, R2
f sub_137BA76		SIR.W	R9, [SP,#0x22C+Var_17C]	f sub_132FDCC	text	:0132913A	STR.W	R9, [SP,#0x22C+var_17C]
f sub_137D17C		BLA.W	j_objc_storestrong_0	f sub_132FDD6	text	:0132913E	BLX.W	jobjc_storeStrong_0
f sub_1380EB8		HUD	R0, SP, #0x22C+Var_150	f sub_132FDE0	text	:01329142	ADD	R0, SP, #0x22C+var_150
f sub_1380EC2	text:0137H244	STD MUVS	K1, #0 D4 [CD #00000.000 4E0]	f sub_132FDF6	text	::01329144	MOUS	R1, #0
f sub_1380ECC		STR	K1, [SP,#0x226+Var_150]	f sub 132FE0C	text	:01329146	STR	R1, [SP,#0x22C+var_150]
f sub_1380ED6			KI, [SF,#0X220+Vdr_178]	f sub 132FE22	text	::01329148	LDR	R1, [SP,#0x22C+var_178]
f sub 1380EE0	Lext:0137H24H	BLA.W	J_UUJC_Storestrong_0	f sub 133071A	text	::0132914A	BLX.W	jobjc_storeStrong_0
7 sub 1380EF6			K0, #0X100 ; S120_L	f sub 133094C	text	:0132914E	MUVW	RU, #UX100 ; S1ZE_t
7 sub 1380F0C	toyt:01970256	MOU	$\frac{1}{100} = \frac{100}{100} = $	f sub 133096C	text	:01329152	BLX.W]malloc_0
f sub 1380E22	toyt:019702EE	000	P1 PC - obje_msgSend_ptr = 0x137H202)	f sub 1220520	text	:01329150	MUV	R1, #(_objc_msgSend_ptr - 0x1329162)
f sub 138181A	toyt:01970260	L DB	P1 [P1] : imp_objc_mcgSond	5 sub_1330E2A	text	:0132915E	HUU	R1, PC ; _oDjC_msgSend_ptr
f sub 138104C	+ ovt - 01970262	MOU	P2 = #(colRof longth - 8v137026E)	J SUD_1550E00	text		LDR	RT, [RT];TMP_ODJC_MSGSend
f cub 1201A6C	toyt:01970260	000	P2 PC - collect longth	7 SUD_1330F80	text	:01329162	MUV	R2, #(SelRef_length - UX132910E)
J SUD_1301A0C	toyt - 01970260	510	DA [SD #Av2204upe 156]	f sub_1331682	text	:0132910H	HUU	RZ, PC ; Selket_length
5 SUD_1501F2A	+ovt - 0137026F	STN EDR	R0 [SP #0x220*Var_124]	f sub_1331A22	text		218	KU, [SF,#UX22C+Var_154]
5 SUD_1381F08	+ovt - 01370270		R9 [R9] • "length"	f sub_1331A2C	text		LDR	кө, [Sr,#0X22C+Var_150] Do [Do] - Шаратты
5 sub_1382080	+ovt • 01370279	STR	R1 [SP #8v22C+02P 188]	f sub_1331A36	text		LDK	KZ, [KZ] ; "IENGTN"
f sub_1382782	+ovt • 01370274	MOU	R1 R2	<u>f</u> sub_1331A4C	text		218	K1, [SF,#0X22C+Var_180]
f sub_1382B22	+ovt - 01370276	I DR	R2 [SP #8v22C+upr 188]	<u>f</u> sub_1331A64	text		MUV	K1, KZ
< 1000D0C		LVn	nz, [3:,#0x220.001_100]	< · · · · · · · · · · · · · · · · · · ·	text	:013291/0	LDK	<pre>k2, [3r,#0x220+Var_180]</pre>
Line 167606 of 23790	0137A1FA 0137A1FA: sub_137	A1FA		Line 166700 of 239361	013290	FA 013290FA: sub_13290	DFA	

Different version, different function name and address, but same code!

Picture Yourself as an Analyst

- You just identified a function of interest
- Questions:
 - Have I seen an equivalent or similar function before?
 - How can I find binaries that contain similar functions?

Security Applications of Code Similarity Detection



Patch analysis

"patch based exploit" generation

Software plagiarism detection



Malware analysis



Existing Works / Tools

Focus on control flow graph (CFG) extraction	 E.g., BinDiff, BinGrep, discovRe, Genius, Gemini
Based on basic block modeling	• E.g., BinHunt, Cop
Run in the emulator	• E.g., BinSim
Rely on Pin tool	• E.g., BLEX, CryptHunt

Existing Works / Tools Limitations



Example (Objective-C)

- 1. (NSString *)encrypt1:(NSString *)message {
- 2. if ([message length] == 0) {
- 3. return @"NULL";
- 4. }
- 5.
- 6. NSString *key = [self makeKey1];
- 7. NSString *encryptedMsg = [self xorWithString:key withMessage:message];

8.

- 9. NSLog(@"encrypt1: %@", encryptedMsg);
- 10. return encryptedMsg;

11. }

Code Obfuscation (IDA decompiled code)

1// ViewController - (id)encrupt1:(id) 2/id cdecl -[ViewController encrypt1:](struct ViewController *self, SEL a2, id a3) 3 3 1// ViewController - (id)encrypt1:(id) 4 struct ViewController *v3; // r5@1 void *v4; // r4@1 5 3 8 int v5; // r1@1 void *vó; // r022 5 int v7: // r0@2 int v8; // r6@2 void *v9; // r0@2 10 int v10; // r5@2 11 12 • 13 v3 = self; • 14 v4 = (void *)objc retain(a3, a2); 14 if (objc msqSend(v4, "length")) • 15 16 -{ • 17 v6 = objc msqSend(v3, "makeKey1"); 18 v7 = objc retainAutoreleasedReturnValue(v6); • 18 19 • 19 v8 = v7: 20 v9 = objc msqSend(v3, "xorWithString:withMessage:", v7, v4); 0 20 21 v26 = self; 22 v27 = a3:• 21 v10 = objc retainAutoreleasedReturnValue(v9); 23 • 22 NSLog(66020, v10); 24 • 23 objc_release(v8); 25 24 -> 26 27 25 else ٤. No Obfuscation 28 26 - { 29 - { 27 v10 = 66004;0 30 • 28 objc retain(66004, v5); 31 29 32 - } 33 0 3 0 objc release(v4); • 34 return (id)j objc autoreleaseReturnValue(v10); 31 35 32 } 36 37 38 • 39 40 <u>41</u> LLVMhShh **Obfuscator** 51 52

2 id cdecl -[ViewController encrypt1:](struct ViewController *self, SEL a2, id a3) int v3; // r6@0 signed int v4; // r001 void *v5; // r0@4 int v6; // r4@4 void *v7; // r0@4 int v15; // r2@14 10 signed int v16; // r1@16 signed int v17; // r2@19 12 signed int v19; // r2@26 13 signed int v22; // r3@28 int v23; // r0@28 15 signed int v24: // r2@30 16 struct ViewController *v26; // [sp+8h] [bp-2Ch]@1 17 id v27; // [sp+10h] [bp-24h]@1 void *v28; // [sp+14h] [bp-20h]@1 struct objc_object *v29; // [sp+18h] [bp-1Ch]@0 objc_retain(a3, a2); v28 = objc msqSend(u27, "length"); v4 = -303701101;while (1) while (1) while (04 > -303701102) if (04 > 1801001627) if (04 != 1801001628) if (04 != 2008731425) goto LABEL 41; v5 = objc_msgSend(v26, "makeKey1"); v6 = objc retainAutoreleasedReturnValue(v5); v7 = objc_msgSend(v26, "xorWithString:withMessage:"); v3 = objc retainAutoreleasedReturnValue(v7); SLog(49496, V3); ic release(v6); goto LABEL_5; objc release(<mark>U27</mark>); objc_autorelease(v29); v4 = -1698061380: else if (04 == -303701101) v4 = 2008731425;53 if (!028)



Code Obfuscation



Obfuscated CFG



Problems

- What features are useful for semantic similarity comparison?
- How these features are captured on mobile platform?
- How to characterize a function with such features?



On-device and Trace-based Similarity Mapping Framework for iOS Apps

Idea

- What features are useful for semantic similarity comparison?
 - dynamic behavior features
- How these features are captured on mobile platform?
 - Dynamic instrumentation
- How to characterize a function with such features?
 - Thread backtrace

Dynamic Behavior Features

We define the concept of "dynamic behavior features" broadly to include any information that can be derived from observations made during execution.

Dynamic Behavior Features

- System calls
 - E.g., read, write, open, etc. (libsystem_kernel.dylib)
- Library calls
 - Memset, memcpy, free. Etc (libSystem.B.dylib)
 - _objc_getClass, _objc_getProtocol, etc. (libobjc.A.dylib)
 - Etc.
- Framework API calls
 - E.g., OpenGLES, CoreMedia, UIKit, etc.

System Design



Preprocessing

• Binary extraction

- Extract app binaries
 - Clutch 2.0
- Remove PIE (Position-independent executable)
 - Armv7: offset 0x401a
 - 0x00 -> 0x01
 - Non-armv7(Arm64): offset 0x18
 - 0x00 -> 0x01
- Re-sign the binary and copy it back to app folder



Preprocessing

- Address extraction
 - Implemented with IDAPython API



Listing 1.1: Function Addresses

```
...
0xb7ea,0xb964,-[VideoSnakeViewController
    toggleRecording:]
0xe2cc,0xe51c,-[VideoSnakeSessionManager
    startRecording]
0x111d8,0x1128c,-[MovieRecorder initWithURL:]
0x1161c,0x116a8,-[MovieRecorder delegate]
0x11834,0x11980,-[MovieRecorder prepareToRecord]
0x11d48,0x11ebc,-[MovieRecorder finishRecording]
...
```

Listing 1.2: Library Addresses

```
0x1606c,__Block_copy
0x1607c,__Block_object_assign
0x1608c,__Block_object_dispose
0x1609c,__Unwind_SjLj_Register
0x160ac,__Unwind_SjLj_Resume
0x160bc,__Unwind_SjLj_Unregister
```

On-device Dynamic Analysis

- Dynamic instrumentation
 - Implemented with FRIDA API: https://www.frida.re/
 - Hook functions
 - Inject JavaScript
- Trace logging
 - Framework API invocations
 - Library function invocations
 - System call invocations
 - Thread backtrace



Listing 1.1: Function Addresses

Listing 1.2: Library Addresses

```
0x1606c,__Block_copy
0x1607c,__Block_object_assign
0x1608c,__Block_object_dispose
0x1609c,__Unwind_SjLj_Register
0x160ac,__Unwind_SjLj_Resume
0x160bc,__Unwind_SjLj_Unregister
```

Listing 1.3: Stack Backtrace: Address



Listing 1.4: Stack Backtrace: Name

```
__Unwind_SjLj_Register,-[MovieRecorder
	prepareToRecord],-[VideoSnakeSessionManager
	startRecording],-[VideoSnakeViewController
	toggleRecording:],sub_B120
_dispatch_get_global_queue,-[MovieRecorder
	prepareToRecord],-[VideoSnakeSessionManager
	startRecording],-[VideoSnakeViewController
	toggleRecording:],sub_B120
_dispatch_async,-[MovieRecorder prepareToRecord],-[
	VideoSnakeSessionManager startRecording],-[
	VideoSnakeSessionManager startRecording],-[
	VideoSnakeViewController toggleRecording:],
	sub_B120
```

. . .

Feature Extraction

• Analyze the stack backtrace information

• Match the library calls/system calls/api calls to corresponding caller functions

Listing 1.5: Function Features

Similarity Searching

- Calculate similarity score
 - Edit distance: sim(fun_x, fun_y)
 - E.g., reference function **fun_x** and target function **fun_y**

Evaluation

• Experiment Setup

- On-device analysis
 - 32GB Apple Jailbroken iPad
 - iOS 8.3
- Feature extraction & similarity searching
 - CPU: Intel Core i7-6700K Processor (8-core with 4.00 GHz)
 - 64 GB memory
 - Ubuntu Linux 14.04 TLS
 - Python 2.7.12
 - IDA Pro 6.6

Evaluation

- Can MobileFindr detect similar functions in different versions of same mobile apps?
- Can MobileFindr detect similar functions used by different mobile apps
- Can MobileFindr be used for analyzing real-world mobile apps?

Ground Truth Dataset I

• 8 Open source iOS app samples:

https://developer.apple.com/library/content/navigation/

Obfuscated by llvm-obfuscator 4.0

- 8 non-obfuscated vs 8 obfuscated version
- Select functions from non-obfuscated version as reference functions
- Keep Debug symbols
 - ground truth through function names

Comparative Results



■ MobileFindr ■ BinGrep ■ BinDiff ■ Genius



- Can MobileFindr detect similar functions in different versions of same mobile apps?
- Can MobileFindr detect similar functions used by different mobile apps
- Can MobileFindr be used for analyzing real-world mobile apps?

Ground Truth Dataset II

- Top used third-part iOS framework
 - AFNetworking 3.0
- 8 opensource projects from GitHub that use the framework above
- Three obfuscation options: (<u>https://github.com/obfuscator-</u> <u>llvm/obfuscator/wiki/Features</u>)
 - Control flow flattening (-fla)
 - Instruction substitution (-sub)
 - Bogus control flow (-bcf)
- Result in 64 apps (56 obfuscated apps + 8 non-obfuscated)
- **Debug symbols -> ground truth through function names**

Comparative Results

	Obfuscation Type
1	NONE
2	-sub
3	-bcf
4	-fla
5	-sub, -bcf
6	-sub, -fla
7	-bcf, -fla
8	-sub, -bcf, -fla

Function Mapping Accuracy 1 0.9 0.8 0.7 0.6 Accuracy 0.5 0.4 0.3 0.2 0.1 0 Top 3 Top 5 Top 10 Top 15 Top K Candidates MobileFindr ■ BinGrep ■ Genius



- Can MobileFindr detect similar functions in different versions of same mobile apps?
- Can MobileFindr detect similar functions used by different mobile apps
- Can MobileFindr be used for analyzing real-world mobile apps?

Analysis Real World Apps

Collect top-ranked apps from iOS App Store

- Compare different versions of each apps
 - E.g., baidu_v9.3 vs baidu_v9.35
- Perform same action for each pair of apps to collect traces
 - E.g., perform web searching with same keywords
- List top 10 similar function candidates
 - Based on the similarity score

Analysis Real World Apps



Function Mapping Accuracy in Top 10

mapping accuracy

Limitations & Discussions

• Path coverage issue

- Incomplete path coverage for dynamic analysis
- May use offloading mechanism for path exploration (presented by Malton, USENIX 2017)
- Small function issue
 - We only evaluate functions contain more than 5 basic block



- Designed a trace-based function similarity mapping system for mobile apps
 - Resilience to various anti-reverse engineering techniques
- Demonstrated the viability of our approach for multiple top-ranked real-world iOS frameworks and apps

THANK YOU!

Q&A?