Instruction Format
[BYTE: Unknown] [BYTE: Instruction] [WORD: Return Value?] [WORD: RDI + 0x34] [WORD: RDI + 0x36 (pointer]

Read watchpoint????

Instruction	RDI + 31h	RDI + 0x3E	RDI + 0x34	RDI + 0x36	Notes	Flag Bytes	
06		4000	4000 (ESI)	0024 (EDX)	Stores data to 0x238120 (Dynamic) + si. Buffer initally stores code (i.e. code + 0x4000). Code is CODEGATE2020{ABCDEFGHIJKL}		
01		Unknown	4f43	b0bd	RDI + 3E = ARG1 + ARG2. (Address 4000,4000)	CO	
04		Unknown	0000	0000	RETVAL = IF RDI[34] == *RDI[0x36] (Compare immediate with indirect)		
05		Unknown	0000	01a0	Compare RETVAL with 0. If it's not zero, do (Unknown)		
01		Standard	4544	babc	Add 4544 to babc. Store it in retval.	DE	
04		Standard	0000	0000			
05		Standard	0000	01a0	Unknown if not 0		
01		Standard	4147	beb9		GA	
04		Standard	0000	0000			
05		Standard	0000	01a0	Unknown if not 0		
01		*4006	4554	baac		TE	
04		*0001	0000	*4006			
05		*0000	*0001	01a0	Unknown if not 0. What is rdi + 0x36?		
01		*4008	3032	cfce			20
04		*0001	0000	*4008			
05		*0000	*0001	01a0			
01		*400a	3032	cfce			20
04		*0001	0000	*400a			
05		*0000	*0001	01a0	Last "Sanity" instruction		
00		*1000 (0)	f974	0000	Saves to 0x1000		
00		*1002 (f974)	2b9d	0000	NEXT KEY		
00		*1004 (2b9d)	4caf	0000		CODEGATE2020{ezpz_	
00	02	*1006 (4caf)	bee1	0000	347d	CODEGATE2020{ezpz_bu	
00	02	*1008 (bee1)	fc0d	0000	5c87	CODEGATE2020{ezpz_but_	
00		*100a (fc0d)	6e48	0000	e589	CODEGATE2020{ezpz_but_1t	
00		*100c	e03c	0000	2e9b	CODEGATE2020{ezpz_but_1t_1	
00		*100e	d322	0000	73ad	CODEGATE2020{ezpz_but_1t_1s_	
00		*1010	1979	0000	94f7	CODEGATE2020{ezpz_but_1t_1s_pr	
00		*1012	36d6	0000	bd19	CODEGATE2020{ezpz_but_1t_1s_pr3t	
00		*1014	40e8	0000	c72b	CODEGATE2020{ezpz_but_1t_1s_pr3t3x}	
00		*1016	cbf7				
00		*0000	dead		First byte is 4. What does this mean?		
00		*0001	0		First byte is 4. What does this mean?		
02	*0002	dead	0002		Constant - returns 0xbd5a		
03	*0000	dead	bd5a		Returns xor dead * bd5a, stores 63f7 in cell 0		
02	*0002	0000	0002		Why? This seems like a "constant multiply", the memory cells aren't impacted by anything befor ethis.		
01	*0154	*1000 = 1000?	*0002 = 0		f974 into 0x154? (0x154 is 0)		
01	***014c**	***400c (417b)**	***0002**		**save LITERAL 400c into 014c (THIS IS SELF MODIFYING)**		
00	*0000	*0x000	*0x0000		I think these are effectively NOP's. I can't see any real instruction save.		
00	*0000	*0x000	*0x0000		**ignore 1 38 (stops here)**		
00	*0000	*0x000	*0x0000				
BREAKPOINT 400c					????		
03	*0002	*400c (657b)	*0000 (63f7)	READ + XOR WITH CONST	68c in 0x2	{e	

01		f974	*0002 (68c)	2b9d is next		
4	02	0	0			
05	FFFF	*0002	1a0	????	Exit if not 0 (exit of {] is not met)	
01	*0001	*0001	01	????		
04	*0002	*0001	*000c		**Compare length if 000c?**	
05	*0001				If not zero, jump to 108.	
02	*0002	*0000 63f7	0002	DOUBLE IT, XOR 2	Should be c7ee. Unknown where it's stored.	
03	*0000	*0000 63f7	0xc7ee (result of last o	DOUBLE, XOR.	Store a419 in slot 0	
01	*0002	01	02		Add 2 to 0x154	
01					Add 2 to 0x14C	
0						
0						
0						
03	*0002	707a	a419		Store d463 into the pile	pz
01		2b9d	d463		Add (then compare to 0)	0x1000

a419