QueryX: Symbolic Query on Decompiled Code for Finding Bugs in COTS Binaries

<u>**HyungSeok Han**</u>^{†*}, JeongOh Kyea[†], Yonghwi Jin[†], Jinoh Kang[†], Brian Pak[†], Insu Yun* [†]Theori Inc. ^{*}KAIST





(Currently postdoc in Georgia Tech)



Static Analysis

• One of the most popular techniques for automatic bug finding

Effective Static Analysis of Concurrency Use-Affec-Free Bugs in Linux Device Devers	20170223 proprint in Surging 2017 prog Profiles and Property Bayes in Jacobiety 2017 prog Profiles and Property State 1 Mark Department of the State of State 1 Mark State of State of State of State 1 Mark	Like & Barrisery Constraints + Laponation, and Economic Amore Default on Introduct Study and Context Amore Default on Advances and context and Advances and Adv	K-Miner: Uncovering Memory Corruption in Linux Intel Gen," these below: Inter their Attendens below? "Strengtheness between themas Geness (and geness and strengtheness). Inter the attendens the	Detecting Missing-Check Bags via Semantic- and Context-Aware Criticalness and Constraints Inferences	Precise and Scalable Detection of Use-after-Compacting-Garbage-Collection Bugs
Bie-He Biel Julia Lawell Qie-Ling Chen, Sh-Min Hu Tahughaa Umversity Sorboane UniversityBuria/LIPS Tahughaa University		ANTEXCT Teaching and the second seco	where the second	Kangjiel Lu, Adrop PARL, and Quark We University of Meancoure	HyungSock Han Andurev Wesle Belan Pak Disort Inc. Theoret Inc. Theoret Inc.
Attract The second sec	 The second second	Strategy with an approximation of the strategy	. They atoms not not good a proof of the second and	DATE The set of the	Image: State in the s

Scalability vs Accuracy

 And a state in the state of the	
-	•

THE DESCRIPTION A COURT OF THE COLD IN AN ADDRESS	which have been holder searcher, and seatland dense. In
services memory is subject to change across these reads, i.e.,	click bypaoing, barter oremox, and contained deputy. In
a race condition, which is known as a double back hog. Price	makey, researchers have expended dealine-jelve ongo to escatale
works have attempted to detect these bags both statically and	privileges on finanzies (15 [11], [12]-
donanically. However, due to their improper assumptions and	What makes also bley/each bag detection an important problem
improduc definitions regarding double-fetch bugs, their multi-	is that, in lernel, it is common to intentionally read data
ben exocute a supervisit surged and sensity from segmental	multiple times from the userspace for performance reasons. We
tabe peerves and tabe negatives, for example, they approach	call this situation a eachi-word. To illustrate, consider fetching
a case of opport terms reacand, and prevents sharply,	a surjuble-length measure with a networkilly recriment size.
and of Budies and double high how from the board of smill	of 4 WB from the second concerned, is to descee out-
much to reasonal perification, which is executer if would at all	ofference of VE before and over 1 VE from the measures in
In this manor, we first present a formal and precise definition of	and a state of the back and copy a the total at the open at
deable-brah hour and then implement a static analysis restern-	one stort resource, in most cases, mis wards incentry and
DUITLDT-to intendedly detect deals-fack logs in OS	CPU cycles if the effective message payload is by bytes or
Leverb, DEADLINE uses shally program analysis includious	not. Hence, the termed handlos this scenario by first telefing a
is systematically find multi-reads throughout the kernel and	4-Byte stee variable and later allocating the buffer and fricking
employs specialized sembolic checking to vet each multi-read for	the size-byte message. A quick scan over the Linux kernel
doutsoonth togs, we apply structure to Linux and FrichSD	reveals that there are over 1,000 multi-reach. Then, a follow-up
someth and find 20 new bugs in Linux and one new bug in	question would be: How many of them are real double-forch
mented double-field have been on one sinch and the doubling	Jury? Until new, the only way to answer this gamtion was to
with bound emissioners	secondly not the consolicated source code of all much mode
	However, this is contained a could become example actions it
1. INTRODUCTION	there are to compare the relevant to the relation to the relation of the
	increase necesses a pressing promens that we have to nots ()
bup is opening system terms can be particularly pro-	screatly define and distinguish double-ferry hope and mark-
tenance. In practice, they often lead to valuerable and that can	such and 2) amounterally verify each work-wal to check
be exploited to compromise the outire system and cause all	whether it is a log.
kinds of severe attacks, such as privilege escalation [1], [2].	Unfortunately, neither aspect has been addressed perfectly in
information leaks [3], and denial of service [4]. This fact has	prior works. Bochspiers [5] defines multi-ready as at least two
drawn serious attention from the security community, and the	memory reads from the same userspace address within a short.
kernel has been increasingly hardened against various types of	time frame, while Wang et al. [11] defines ender-reads haved
memory array, e.g., kASLR 19, kCFI (6), 171, and UniSan (\$1	on a few empirical static code nations. Due to the immediate
Hadinational de chara militari insu hana Kanita Lenerare in tamine	definitions had made much is more false mailing the
whether that market have been	increased a identified based and false specifier (i.e. printer)
then does of hode how due has secondly doesn standard in	here). Here hereated, adden of then one contend offic
the case of type regritter for meeting teacher and the	supremented and the second second second second second
senser your rays, warm in another project [1] introduced	wangene ware your only two onto ones in dependen-
for the Windows kernel. Wang et al. also studied should get the	and they completely lower it to manual verification.
bugs for the Linux Jernel [10]. A double detablishing is a special	In this paper, we propose DEADLINE, an automatic local to
type of nee condition bug in which (typically during syscall	statically detect multi-reads and double-feach hugs with both
encention) the kernel reads a particular usenpace memory	high precision and coverage. In particular, DEADRAME covers
region more than once with the assumption that the content	all drivers, file systems, and other peripheral modules that can
in the accessed region does not chance acress reads. Hewever,	be compiled under the x86 architecture for both Linux and
this approaching is not colid. A composite his complete case through	the free \$50 baseds. Diricht put as decoursed all site solated
and hereinghold in the rank of concentration in histories. Install	due Ma faith from semanted in LNN and further found 71 years
can be and the second second second	reserves on the second of the second
9 2018, Meng, Su, Under Jopmeno REE. 56	discounter.
LAN IN TRANSPORTATION OF THE TRANSPORT	Sobety

Abstract	is a write, and the relative ordering of the two accesses is
We present the for a state agreemb for properties of the state of the	The effective part of the theorem (1) and (1) of the effective part of the theorem (1). It is also that the effective part of the theorem (1) and (1) and (1) and (1) and (1) and (1) and (1)
1 Introduction	their analysis cannot cover the entirety of the kernel. In fact, their supressed has not discovered any dischlocked
The wide use of multi-row hards are in making occur- ent programs incomingly provider, experially in spe- roling yourne, real-time systems and comparing inste- olec systems. However, ensembler in the system of the needed sector systems of the system of the system work of concernersy bargs can be comparing into their speed concernersy bargs can be comparing into their system concernersy bargs can be comparing into their system concerners (1990).	vulnerability in Linux, Peuel/SD or Opeu/SD [11], Bei docs, Arezyk and Oshbrinh Inner brought attention to not only on how to find her about n-how is exploit another firsth-vulnerabilities, Instructions on how to exploit aloue her forkels interv security become guided yourability [11]. Thans, anding intereds, in particulate drivers, for instability fields vulnerabilities has become urgest. Decise about no see existed in struct for the more here 1 particulation see existed in the secure under the securement the
A data and is perfore common citation is commont	bridge hardware and software by previding interfaces be

And are appreciable to approximation of the second	A set of the set of th
- montan	for users to investigate further. In this case, the error is le entited, so that the execution can continue. (3) Passing error

renergy conditions. Mereover, moders Arlo, solves and the second	products Respired Weights productions on solu- dies ATRNS with the constraints and a fait of Theorem Theorem 2000 and the solution of the solution of the solution of the solution of the solution of the solution of the solution of the solution of the solution of the solution of the solution of the solution of the solution which are a solution of the solution of the solution which are a solution of the solution o	
Introduction	contradictions in source code is that it achieves preci- sion by considering semantic constraints in API usage	
b, logs and couples unliver is hold with more more integrated out ANN. While ANN neuropowers integrated out ANN. While ANN neuropowers the integrated integrates and the state of the state of the state of the state of the state of the state of the state one coversely implementing. For example, unside of ANN of an order of the state of the state of the state of the ANN of the state of the state of the state of the state of the ANN of the state of the state of the state of the ANN of the state of the state of the state of the state of the ANN of the state of the state of the state of the ANN of the state of the state of the state of the state of the state of the state of the state of the state of the state of the states, which we generally on the state of the states of the states of the state of the st	syndrific concepts that in comparison and a syndrific ora- center hand to arthrogen. The technique, could induced concludie concerts, we chorecreate the head photon parts and the syndrometry of the syndrometry of the syndrometry of informing unsurantic heads. In a syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of the syndrometry of	
to as APIs in this paper. ary different tools, techniques, and methodologies been proposed to address the problem of finding- recenting API usage errors. Breadly, all existing ignes other require (1) menual effort—API specific	AP specific liceoxidege for more precise statistic and deeper security analysis. We develop the spin sectors in APESAN that are tailmed to check a variety of properties with accurity implications, such as cryptographic prot- ected API moreors, singer verifics, impreper locking, and	

6.5 hours and identified 264 previously unknown memory storp hugs—badge to 76 CVRs and 128 RatKet advised being Black, which represent 51.6% of memory safety logg- properts to RatKets size 2005. The weee being RURA. A found are non-critical, under, and alten made by Rata togenter two in the Rata standard beney, one in the wee being RURA. Some and one in the Rata complete. RURA is is open-source, and prove of to algorithm is integrated non-to efficial flucture.	memory-soft very by default. This unique paradigm while provide both using valent orderation, ratice Rara tappeals for developing system offware. Bark has stated to recei- miger adoption is conventional system software and as op- ning systems (22, 33, 36, 47), embedded systems (26), we frameworks, [21] and with horsewire [59), when both accur- ant parternance are independed. The key idea of Har's memory andery is to validate the The key idea of Har's memory andery is to validate the
CCS Concepts: * Theory of computation \to Program analysis; * Security and privacy \to Suftware and application security.	valuates the access and the lifetime of memory-allocated or jects (or values). Simply put, each value in Rear has an own variable, and the memory used for the value is immediate
Keywords: Bast, Memory-safety, Program analysis	reclaimed when the owner variable goes out of scope. Rast ownership system is often viewed as similar in concept to up
1 Introduction	structural type systems [61, 62] but supports a novel concep
Rost is an emerging programming language for system soft- wate. As a system language like C or C++, its primary concern is to enable nurive performance and to allow programmers	of howevering that allows the creation of shared or matali- references to values. The compiler's hornw checker provide two guarantees: 1) references cannot outlive their owner va- ables remember was after-free (1MF) veherschilten and
Promission to make digital at hard cogine of 61 or part of this work for promotine or known one is ginarit arbitraril program of the open ranks of the antice of the second second second second second second the notice and the full collesies on the first page. Cogregiths for comparison of the second second first of the second second second second second second second second second second second second collesis permitted. To copy selections or rapidlish, to post an array second seco	both shared and reachile references are rever present at 6 same time, elimitating the possibility of concentre real at applies to the value (Figure 3). Ultiferramatic), each adray adae are often two nextriciv- in certain system software fuel queues low-level bardware socies (σ_{eff} , according true pointer) of bandper performan- tion in protocol (σ_{eff}) and σ_{eff} or the present of σ_{eff} contains the socies (σ_{eff} , according to the pointer) of σ_{eff} or the pre- tain algority fractional pre- position (σ_{eff}) are contained to your excitation by addi Rash true are constrained to your outpress or software. Rull to turbulence the concept of analysis, in which the dary of the

Extensible Static Checking Tools

• Static analysis + Domain knowledge (Query)



Extensible Static Checking Tools

• Static analysis + Domain knowledge (Query)

Binary ????

	Input	Analysis objects	Syntactic	Data-flow	Symbolic
CodeQL [22] joern [76] Sys [7]	Source Source Source	Source Source LLVM IR	\checkmark	\checkmark	× × √

Extensible Static Binary Checking Tools

• Query based on **binary IRs**



Extensible Static Binary Checking Tools

• Query based on **binary IRs**



Inconsistency b/w Analyst & Query

- Analysts mostly work with **decompiled code** due to its high-level information such as high-level control flows and types.
- But, queries are based on **binary IRs** because binary IRs are more analyzer-friendly.

<pre>int64fastcall isPalindrome(const char int i; // [rsp+18h] [rbp-8h] int v3; // [rsp+1Ch] [rbp-4h]</pre>	*a1) {
<pre>v3 = strlen(a1); for (i = 0; i < v3 / 2; ++i) { if (a1[i] != a1[v3 - i - 1]) return 0LL; } return 1LL;</pre>	
}	

Extensible Static Binary Checking Tools

• Static analysis + Domain knowledge (Query)



The Convergence of Source Code and Binary Vulnerability Discovery - A Case Study (AsiaCCS '22)

- Feed decompiled code from Hex-Rays to CodeQL and joern.
- Compared to tools with the original source code, tools with decompiled code got less TP and more FP because..

```
int glob_x = 0x42;
                                 int main() {
struct Pos { int x; int y; };
                                   int v1; // [rsp+0h] [rbp-20h]
int main() {
                                   int v2; // [rsp+4h] [rbp-1Ch]
  struct Pos pos;
                    X: 0x42
                                                    X: 0x42
  pos.x = glob_x;
                                   v1 = dword_201
                    Y: 0x43
                                                    Y: ????
  pos.y = 0x43;
                                   v2 = 0x43;
  print_pos(&pos);
                                   print_pos(&v1);
         Original code
                                          Decompiled code
```

The Convergence of Source Code and Binary Vulnerability Discovery - A Case Study (AsiaCCS '22)

- Feed decompiled code from Hex-Rays to CodeQL and joern.
- Compared to tools with the original source code, tools with decompiled code got less TP and more FP because..

Previous source code analysis tools are binary-unaware!

Symbolic QueryX: Query on Decompiled Code

• Static analysis + Domain knowledge (Query)

	Input	Analysis objects	Syntactic	Data-flow	Symbolic
CodeQL [22] joern [76] Sys [7]	Source Source Source	Source Source LLVM IR	\checkmark	\sim	× × ✓
angr [<mark>56</mark>] BAP [<mark>9</mark>]	Binary Binary	VEX IR BIL	\checkmark	\checkmark	\checkmark
QUERYX	Binary	Decompiled code	\checkmark	\checkmark	\checkmark

Goals & Approaches of QueryX

- 1. Binary-aware analysis on decompiled code
 - Analysis based on our new IR, DNR
- 2. Analyst-friendly symbolic query
 - Symbolic query based on decompiled code and callbacks
 - JavaScript-like query
- 3. Scalable analysis with analyst-friendly symbolic query
 - Under-constrained symbolic execution
 - CFG reduction based on callbacks and their dependencies
 - => Check our paper

QueryX Architecture



Goals & Approaches of QueryX

- 1. Binary-aware analysis on decompiled code
 - Analysis based on our new IR, DNR
- 2. Analyst-friendly symbolic query
 - Symbolic query based on decompiled code and callbacks
 - JavaScript-like query
- 3. Scalable analysis with analyst-friendly symbolic query
 - Under-constrained symbolic execution
 - CFG reduction based on callbacks and their dependencies





<pre>int main() { int v1; // [rsp+0h] [rbp-20h] int v2; // [rsp+4h] [rbp-1Ch]</pre>	<pre>stack = Alloc(0x8) // v1: @(stack + 0x0) // v2: @(stack + 0x4)</pre>
<pre>// ISSUE1: Binary-embedded data v1 = dword_2010A4; v2 = 0x43;</pre>	<pre>Store(stack, Load(ProgAddr(0x2010a4), 4), 4)</pre>
<pre>//ISSUE2: Binary-dependent code print_pos(&v1); }</pre>	Call (print_pos, [stack])



Goals & Approaches of QueryX

- 1. Binary-aware analysis on decompiled code
 - Analysis based on our new IR, DNR
- 2. Analyst-friendly symbolic query
 - Symbolic query based on decompiled code and callbacks
 - JavaScript-like query
- 3. Scalable analysis with analyst-friendly symbolic query
 - Under-constrained symbolic execution
 - CFG reduction based on callbacks and their dependencies



```
for (func in prog.functions)
  symExec(func, symRule);
                              Perform symexec from the entry of
                                   func based on symRule
function symRule (node) {
  if (isCall(node, "malloc"))
    setCallback(node, collectAlloc, "alloc");
  if (isCall(node, "memcpy"))
    setCallback(node, checkMemcpy, "copy");
```







<pre>function collectAlloc (node, state) {</pre>
<pre>// node: AST node of malloc call</pre>
<pre>// state: Current symbolic state</pre>
<pre>var addr = state.getValue(node);</pre>
<pre>var size = state.getValue(node.args[0]);</pre>
<pre>if (state.allocs == undefined) state.allocs = [];</pre>
<pre>// Store allocated address and its size</pre>
<pre>state.allocs.push({addr: addr. size: size});</pre>
2



```
for (func in prog.functions)
  symExec(func, symRule);
function symRule (node) {
  if (isCall(node, "malloc"))
    setCallback(node, collectAlloc, "alloc");
  if (isCall(node, "memcpy"))
    setCallback(node, checkMemcpy, "copy");
}
function collectAlloc (node, state) {
  // node: AST node of malloc call
  // state: Current symbolic state
  var addr = state.getValue(node);
  var size = state.getValue(node.args[0]);
  if (state.allocs == undefined) state.allocs = [];
  // Store allocated address and its size
  state.allocs.push({addr: addr, size: size});
function checkMemcpy (node, state) {
  var dst = state.getValue(node.args[0]);
 var size = state.getValue(node.args[2]);
  for (var alloc of state.allocs) {
    if (dst.includes(alloc.addr)
      && state.isSAT(alloc.size < size)) {</pre>
      print("Overflow detected");
```

 A query for finding heap overflow due to integer overflow in Windows kernel



CVE-2021-31979 (Heap overflow due to integer overflow)

NTSTATUS	fastcall	<pre>NtfsSetDispositionInfo()</pre>	{
----------	----------	-------------------------------------	---

length = volumeName->Length + dirName->Length

```
+ fileName->Length;
```

. . .

(1)

Idx	CVE	Program	Function	Bug Type	Bounty
1	CVE-2021-41370	ntfs.sys	NtfsSetShortNameInfo	Heap overflow	\$20,000
2	CVE-2021-41378	ntfs.sys	NtfsSetDispositionInfo	Heap overflow	\$20,000
3	CVE-2021-43229	ntfs.sys	TxfAllocateAndStoreNameForTxfLogging	Heap overflow	\$20,000
4	CVE-2021-43230	ntfs.sys	TxfAllocateFullFilePathForChangeNotify	Heap overflow	\$20,000
5	CVE-2021-43231	ntfs.sys	NtfsRenameToPrivateDir	Heap overflow	\$20,000
6	CVE-2021-41367	ntfs.sys	TxfOpenFileProcessing	Heap overflow	\$20,000
7	CVE-2022-23293	fastfat.sys	FatSetFullNameInFcb	Heap overflow	\$20,000

CVE-2021-41378, one of heap overflow bugs QueryX found

angr vs QueryX in the Example



angr: 267 LoC, QueryX: 33 LoC

def collect_args (state, addr):
 state.queryx.args = [state.regs.rcx, state.regs.rdx, state.regs.r8]

def hook_alloc (state, addn):
 if len(state.queryx.args) == 0: return
 size = state.queryx.args[1]
 state.queryx.args[1]
 state.queryx.allocs.append((state.regs.rax, size, state.addr))
 state.queryx.args = []

def hook_memmove (state, addr): if len(state.queryx.args) == 0: return dst = state.queryx.args[0]

var dst = state.getValue(node.args[0]); var size = state.getValue(node.args[2]); for (var alloc of state.allocs) { if (dst.includes(alloc.addr) && state.isSAT(alloc.size < size)) { print("Overflow detected"); } }

Evaluation (1-day)

 Comparison on the Dataset of Mantovani *et al.* ("The convergence of source code and binary vulnerability discovery–a case study", AsiaCCS '22)

		QUERYX		Moro TD			joern	CodeQL
Bug		Total	Detected			tected	Detected	Detected
CVE-2017-1000249	5s	1	\checkmark	3m	1	\checkmark	✓	✓
CVE-2013-6462	30s	1	\checkmark	6 s	2	\checkmark	\checkmark	\checkmark
BUG-2012	-	-	×	-	-	×	×	×
CVE-2017-6298	4s	1	\checkmark	24s	1	\checkmark	\checkmark	\checkmark
CVE-2018-11360	13h	13	\checkmark	> 24h	303	\checkmark	×	×
CVE-2017-17760	2.5m	19	\checkmark	1.8h	30	\checkmark	\checkmark	×
CVE-2019-19334	40s	1	\checkmark	1.1h	1	\checkmark	\checkmark	\checkmark
CVE-2019-1010315	20m	4	\checkmark	13m	4	×	\checkmark	\checkmark
BUG-2010	3 s	1	\checkmark	12m	2	\checkmark	×	×
BUG-2018	2.6h	2	\checkmark	> 24h	18	\checkmark	\checkmark	×

Evaluation (1-day)

• Comparison on the Dataset of Mantovani, *et al.* ("The convergence of Lack of high-level info discovery-a case student (e.g., type)

Binary-unaware

	QUERYX			angr			joern	CodeQL
Bug	Time	Total	Detected	Time	Total	Detected	Detected	Detected
CVE-2017-1000249	5 s	1	\checkmark	3m	1	\checkmark	√	✓
CVE-2013-6462	30s	1	\checkmark	<u>6s</u>	2	\checkmark	\checkmark	\checkmark
BUG-2012	-	-	×	-	-	×	×	×
CVE-2017-6298	4s	1	\checkmark	24s	1	\checkmark	\checkmark	\checkmark
CVE-2018-11360	13h	13	\checkmark	> 24h	303	\checkmark	×	×
CVE-2017-17760	2.5m	19	\checkmark	1.8h	30	\checkmark	\checkmark	×
CVE-2019-19334	40s	1	\checkmark	1.1h	1	\checkmark	\checkmark	\checkmark
CVE-2019-1010315	20m	4	\checkmark	13m	4	×	\checkmark	\checkmark
BUG-2010	3 s	1	\checkmark	12m	2	\checkmark	×	×
BUG-2018	2.6h	2	\checkmark	> 24h	18	\checkmark	\checkmark	×

Evaluation (0-day)

 QueryX found 15 previous unknown vulnerabilities including 10 CVEs and earned \$180,000 from MS bug bounty program.

Idx	CVE	Program	Function	Bug Type	Bounty
1	CVE-2021-41370	ntfs.sys	NtfsSetShortNameInfo	Heap overflow	\$20,000
2	CVE-2021-41378	ntfs.sys	NtfsSetDispositionInfo	Heap overflow	\$20,000
3	CVE-2021-43229	ntfs.sys	TxfAllocateAndStoreNameForTxfLogging	Heap overflow	\$20,000
4	CVE-2021-43230	ntfs.sys	TxfAllocateFullFilePathForChangeNotify	Heap overflow	\$20,000
5	CVE-2021-43231	ntfs.sys	NtfsRenameToPrivateDir	Heap overflow	\$20,000
6	CVE-2021-41367	ntfs.sys	TxfOpenFileProcessing	Heap overflow	\$20,000
7	CVE-2022-23293	fastfat.sys	FatSetFullNameInFcb	Heap overflow	\$20,000
8	CVE-2022-30162	win32kfull.sys	NtUserSetClassLongPtr	Kernel Address Disclosure	-
9	CVE-2020-17041	PrintConfig.dll	CopyFileFromPrinterData	Path Traversal	\$20,000
10	CVE-2020-17042	PrintConfig.dll	UniDrvUI::PConcatFilename	Path Traversal	\$20,000
11 – 15	-	Automotive	REDACTED	Out-of-bound Access	-

More in the paper

- Scalable analysis with analyst-friendly symbolic query
- 4 kinds queries for finding 0-days
- More detail comparison against other tools
- Other evaluations and details

Thank you