CodeQL Java Optimisation

Make CodeQL Data Flow Analysis support java features such as java reflection, threading, etc.

演讲人：m0d9

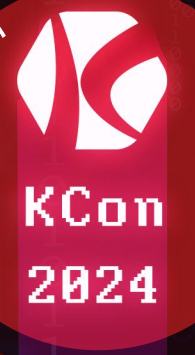
时间：2024.08.24

关于我

- ID：m0d9
- From: 腾讯云云鼎实验室

目录

CONTENT



KCon

PART ONE

01

CodeQL 概述与Java 分析难点

CodeQL 概述

- CodeQL 历史

- 程序分析
  - PTA (soot/Tai-e)
  - Datalog(Soufflé/Doop)

CodeQL 概述

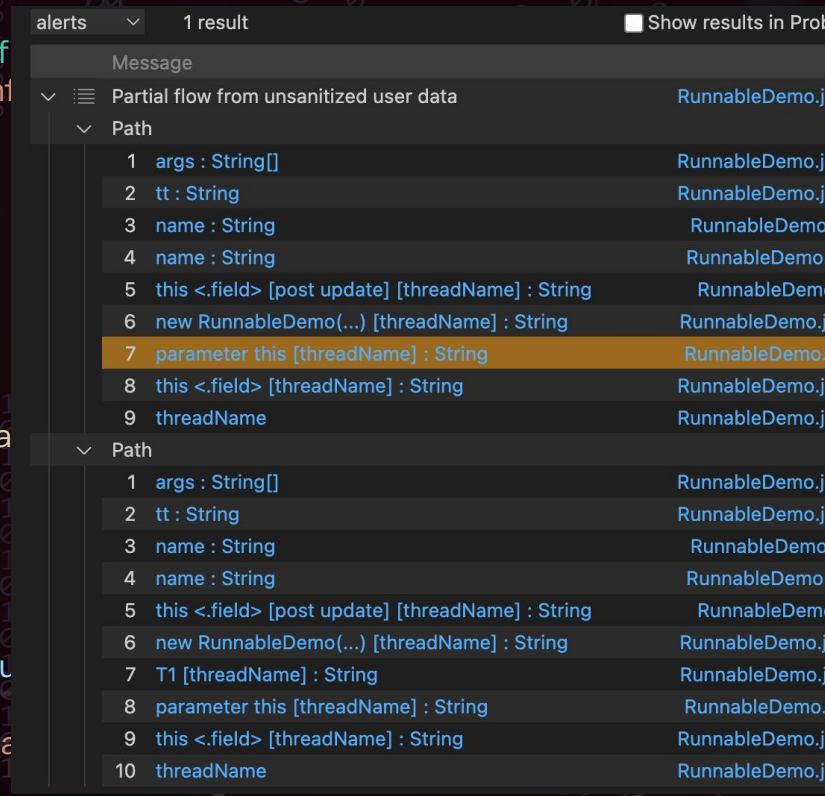
Demo

KCon

Source Code + QL => Result

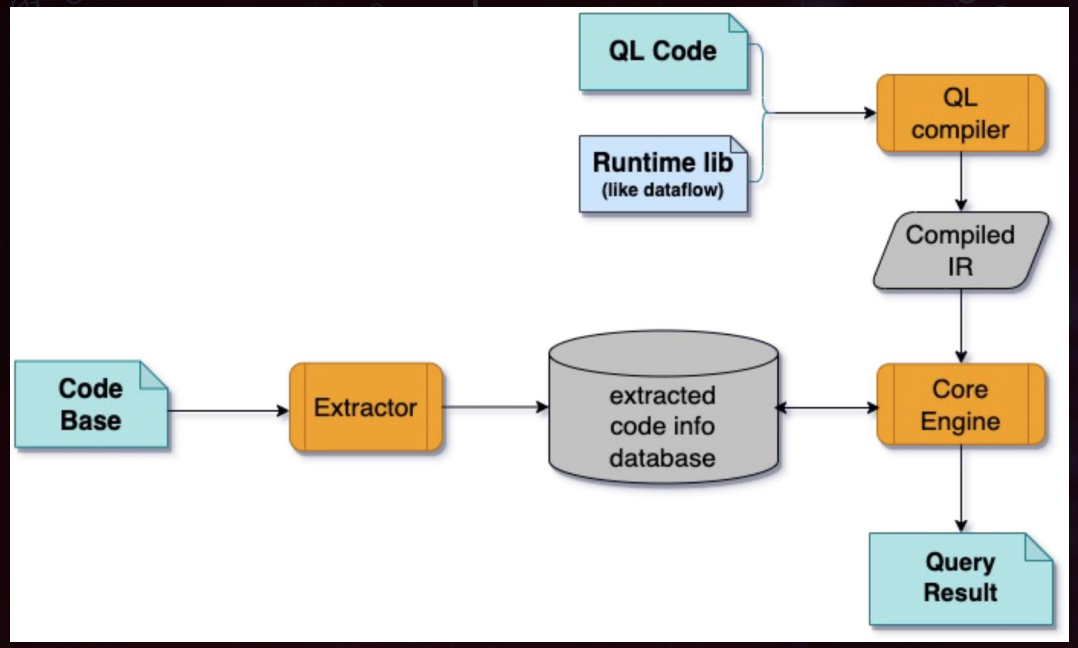
```
import java.lang.Runnable;

public class RunnableDemo implements Runnable {
    private String threadName;

    RunnableDemo(String name){
        threadName = name;
    }

    public void run(){
        System.out.println(threadName);
    }

    public static void main(String[] args) throws Exception {
        String tt = args[0];
        RunnableDemo T1 = new RunnableDemo(tt);
        Thread t = new Thread(T1);
        t.start();
    }
}
```

```
class MyTaintTrackingConfig extends TaintTracking::Conf
  MyTaintTrackingConfig() { this = "MyTaintTrackingCont
  override predicate isSource(DataFlow::Node source) {
      exists(Method m |
          m.hasName( "main")
          and m.getAParameter() = source.asParameter()
      )
  }
  override predicate isSink(DataFlow::Node sink) {
      exists(MethodAccess ma |
          ma.getCallee().getDeclaringType().hasQualifiedNa
          sink.asExpr() = ma.getAnArgument()
      )
  }
}

from MyTaintTrackingConfig cfg, DataFlow::PathNode sou
where cfg.hasFlowPath(source, sink)
select sink.getNode(), source, sink, "Partial flow from unsa
```



CodeQL 概述

架构



- DB 类似DataLog 的facts，有自己的格式
- QL 编译形成dil，dil 类似DataLog的DL

CodeQL 概述

Database

KCon

source code → .trap → .rel (dbscheme)

RunnableDemo.java

test > query-tests > multi-thread3 > RunnableDemo.java

```
1  import java.lang.Runnable;
2
3
4  public class RunnableDemo implements Runnable {
5      private String threadName;
6
7      RunnableDemo(String name){
8          threadName = name;
9      }
10
11     public void run(){
12         System.out.println(threadName);
13     }
14
15     public static void main(String[] args) throws Exception {
16         String tt = args[0];
17         RunnableDemo T1 = new RunnableDemo(tt);
18         Thread t = new Thread(T1);
19         t.start();
20     }
21 }
22
23
24
```

RunnableDemo.java.trap

test > query-tests > multi-thread3 > multi-thread3.testproj > trap > java > opt > codeql-ho

```
407  #10137=@"callable;{#10125}.<init>({#10011}){#10016}"
408  callableBinding(#10131,#10137)
409  variableBinding(#10135,#10114)
410  #10138=*
411  stmts(#10138,14,#10081,3,#10079)
412  #10139=*
413  locations_default(#10139,#10000,19,9,19,18)
414  hasLocation(#10138,#10139)
415  #10140=*
416  exprs(#10140,61,#10016,#10138,0)
417  callableEnclosingExpr(#10140,#10079)
418  statementEnclosingExpr(#10140,#10138)
419  #10141=*
420  locations_default(#10141,#10000,19,9,19,17)
421  hasLocation(#10140,#10141)
422  #10142=*
423  exprs(#10142,60,#10125,#10140,-1)
424  callableEnclosingExpr(#10142,#10079)
425  statementEnclosingExpr(#10142,#10138)
426  #10143=*
427  locations_default(#10143,#10000,19,9,19,9)
428  hasLocation(#10142,#10143)
429  #10144=@"callable;{#10125}.start(){#10016}"
430  callableBinding(#10140,#10144)
431  variableBinding(#10142,#10129)
432
```

semmlecode.dbscheme | callableBinding.rel

query-tests > multi-thread3 > multi-thread3.testproj > db-java > default > callableBinding.rel

HEX

```
          00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  已解码的文本
00000000  00 00 6A 47 00 00 11 4F 00 00 6A 58 00 00 04 C9  . . j G . . . O . . j X . . .
00000010  00 00 6A 81 00 00 00 20 00 00 6A 8F 00 00 16 6D  . . j . . . .  . . j . . . m
00000020  00 00 6A 97 00 00 16 4E +                        . . j . . . . N +
```

CodeQL 概述

QL

KCon

.ql → .dil → .ra

.dil

Datalog intermediate representation

.ra

Relational algebra intermediate representation

test > query-tests > multi-thread3 > patched.ql > { } patched > MyTaintTrackingCon

```
1   /**
2    * @kind path-problem
3    */
4   import java
5   import semmle.code.java.dataflow.DataFlow
6   import semmle.code.java.dataflow.TaintTracking
7   import DataFlow::PathGraph
8
9   class MyTaintTrackingConfig extends TaintTracking::Configuration {
        Quick Evaluation: MyTaintTrackingConfig
10      MyTaintTrackingConfig() { this = "MyTaintTrackingConfig" }
11
        Quick Evaluation: isSource
12      override predicate isSource(DataFlow::Node source) {
13          exists(Method m |
14              m.hasName( "main")
15              and m.getAParameter() = source.asParameter()
16          )
17      }
18
        Quick Evaluation: isSink
19      override predicate isSink(DataFlow::Node sink) {
20          exists(MethodAccess ma |
21              ma.getCallee().getDeclaringType().hasQualifiedName("java.i
22              sink.asExpr() = ma.getAnArgument()
23          )
24      }
25  }
26
27  from MyTaintTrackingConfig cfg, DataFlow::PathNode source, DataFlow::P
28  where cfg.hasFlowPath(source, sink)
29  select sink.getNode(), source, sink, "Partial flow from unsanitized us
30
31
```

test > query-tests > multi-thread3 > patched.dil

```
155424      ),
155425      exists(string arg1, string arg2 |
155426        arg1 = "java.util",
155427        arg2 = "Collection",
155428        Type::RefType::hasQualifiedName#dispred#f0820431#fff(call_result, arg1,
155429                                                             arg2)
155430      )
155431    )
155432  .
155433
155434  DataFlowImpl::Configuration::isSink#dispred#f0820431#ff(/* DataFlowImpl::Configura
155435                                                          /* DataFlowNodes::Public::
155436  :-
155437    exists(/* Expr::Expr */ cached entity call_result#3 |
155438      exists(/* Expr::MethodAccess */ cached entity ma |
155439        exists(/* Type::RefType */ cached entity call_result |
155440          exists(/* Member::Callable */ cached entity call_result#2 |
155441            this = "MyTaintTrackingConfig",
155442            Expr::Call::getCallee#dispred#f0820431#ff(ma, call_result#2),
155443            Member::Member::getDeclaringType#dispred#f0820431#ff(call_result#2,
155444                                                                call_result)
155445          ),
155446          exists(string arg1, string arg2 |
155447            arg1 = "java.io",
155448            arg2 = "PrintStream",
155449            Type::RefType::hasQualifiedName#dispred#f0820431#fff(call_result,
155450                                                                 arg1, arg2)
155451          )
155452        ),
155453        Expr::MethodAccess::getAnArgument#dispred#f0820431#ff(ma, call_result#3)
155454      ),
155455      DataFlowNodes::TExprNode#b728817f(call_result#3, sink)
155456    );
155457    exists(/* Expr::Expr */ cached entity call_result#2 |
```

test > query-tests > multi-thread3 > patched.ra

```
40275   EVALUATE NONRECURSIVE RELATION:                                    L
40294     {4} r10 = JOIN r9 WITH Expr::ConstructorCall::getConstructedType#dispred#f0820
40295     {3} r11 = JOIN r10 WITH Intent::TypeIntent#class#f5dac5da#f ON FIRST 1 OUTPUT
40296     {3} r12 = JOIN r11 WITH Expr::Expr::getType#dispred#f0820431#ff ON FIRST 1 OUT
40297     {3} r13 = JOIN r12 WITH Intent::TypeIntent#class#f5dac5da#f ON FIRST 1 OUTPUT
40298     {2} r14 = JOIN r13 WITH Expr::ClassInstanceExpr::getArgument#dispred#f0820431#
40299     {2} r15 = JOIN r14 WITH DataFlowNodes::TExprNode#b728817f#ff ON FIRST 1 OUTPUT
40300
40301     {2} r16 = r7 UNION r15
40302     return r16
40303
40304   EVALUATE NONRECURSIVE RELATION:
40305     DataFlowImplForOnActivityResult::Configuration::isSink#dispred#f0820431#cpe#2#f(
40306       SENTINEL RandomDataSource::RandomDataSource::getOutput#dispred#f0820431#ff#sha
40307       SENTINEL Expr::MethodAccess::getMethod#dispred#f0820431#ff
40308       SENTINEL DataFlowNodes::TExprNode#b728817f#ff
40309       SENTINEL Member::Member::getDeclaringType#dispred#f0820431#ff
40310       SENTINEL OnActivityResultSource::ActivityOrFragment#class#0eb85844#f
40311       {3} r1 = JOIN RandomDataSource::RandomDataSource::getOutput#dispred#f0820431#f
40312       {2} r2 = JOIN r1 WITH Element::Element::hasName#dispred#f0820431#ff ON FIRST 2
40313       {2} r3 = JOIN r2 WITH DataFlowNodes::TExprNode#b728817f#ff ON FIRST 1 OUTPUT L
40314       {2} r4 = JOIN r3 WITH Member::Member::getDeclaringType#dispred#f0820431#ff ON
40315       {2} r5 = JOIN r4 WITH Type::hasDescendant#6144c3fd#ff_10#join_rhs ON FIRST 1 O
40316       {1} r6 = JOIN r5 WITH OnActivityResultSource::ActivityOrFragment#class#0eb8584
40317       return r6
40318
40319   EVALUATE NONRECURSIVE RELATION:
40320     SYNTHETIC Expr::ConstructorCall::getConstructedType#dispred#f0820431#ff_10#join_
40321       SENTINEL Expr::ConstructorCall::getConstructedType#dispred#f0820431#ff
40322       {2} r1 = SCAN Expr::ConstructorCall::getConstructedType#dispred#f0820431#ff OU
40323       return r1
40324
40325   EVALUATE NONRECURSIVE RELATION:
40326     DataFlowImplForOnActivityResult::Configuration::isSource#dispred#f0820431#cpe#2#
40327       SENTINEL Expr::ConstructorCall::getConstructedType#dispred#f0820431#ff_10#join
```

Why hard language features are hard to analyze?

- Java Reflection
- Native Code

Java 程序分析难点

Runnable

```
import java.lang.Runnable;


public class RunnableDemo implements Runnable {
    private String threadName;

    RunnableDemo(String name){
        threadName = name;
    }

    public void run(){
        System.out.println(threadName);
    }

    public static void main(String[] args) throws Exception {
        String tt = args[0];
        RunnableDemo T1 = new RunnableDemo(tt);
        Thread t = new Thread(T1);
        t.start();
    }
}
```

- java.lang.Runnable
- java.lang.Thread
- java.util.concurrent.Callable
- java.util.concurrent.FutureTask
- java.util.concurrent.ExecutorService
- lambda
- …

Java 程序分析难点

Reflection

KCon

```java
import java.lang.reflect.Method;

public class InvokeDemo {
    private String name;

    public void setName(String name){
        this.name = name;
    }

    public String getName(){
        return name;
    }

    public static void main(String[] args) throws Exception {
        String content = args[0];
        InvokeDemo tt = new InvokeDemo();

        Method method11 = tt.getClass().getMethod("setName", String.class);
        method11.invoke(tt, content);

        String name = tt.getName();
        System.out.println(name);
    }
}
```

- newInstance
- getMethod
- invoke
- set/get
- …

Java 程序分析难点

其他

✓ Lombok: CodeQL v2.14.4 已解决

- https://github.blog/changelog/2023-09-01-code-scanning-with-codeql-improves-support-for-java-codebases-that-use-project-lombok/

✓ 非源码DB构建: v2.16.5 已解决

- https://github.com/github/codeql-cli-binaries/blob/HEAD/CHANGELOG.md#release-2165-2024-03-21

？ 其他未知...

KCon

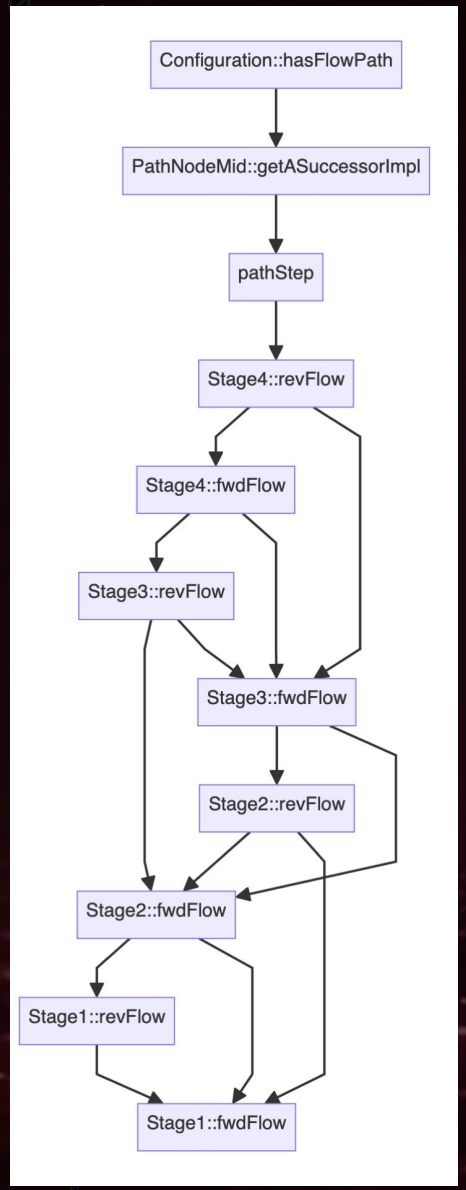
PART ONE

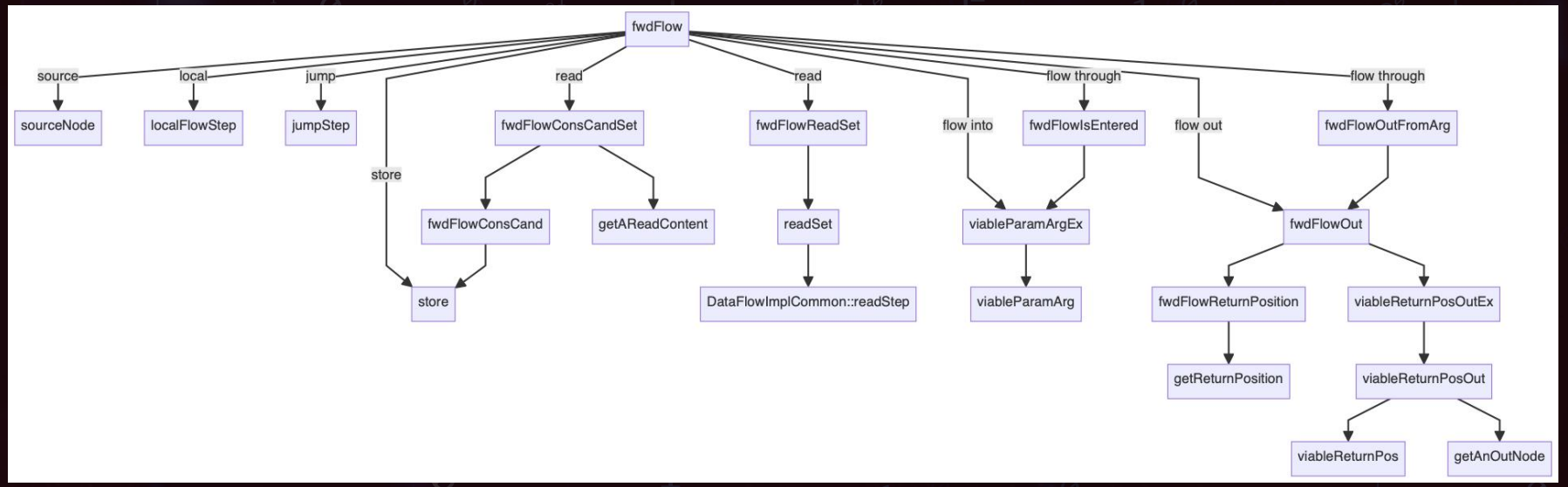
02

CodeQL 数据流分析解析

CodeQL DataFlow分析

DFA





CodeQL DataFlow分析

DataFlow::Node

- TExprNode(Expr e)
- TExplicitParameterNode(Parameter p)
- TImplicitVarargsArray(Call c)
- TInstanceParameterNode(Callable c)
- TImplicitInstanceAccess(InstanceAccessExt ia)
- TMallocNode(ClassInstanceExpr cie)
- TExplicitExprPostUpdate(Expr e)
- TImplicitExprPostUpdate(InstanceAccessExt ia)
- TFlowSummaryNode(FlowSummaryImpl::Private::SummaryNode sn)
- TFieldValueNode(Field f)
- TCaptureNode(CaptureFlow::SynthesizedCaptureNode cn)
- TAdditionalNode(Expr e, string id)

test > query-tests > multi-thread3 > RunnableDemo.java

```
1  import java.lang.Runnable;
2
3
4  public class RunnableDemo implements Runnable {
5      private String threadName;
6
7      RunnableDemo(String name){
8          threadName = name;
9      }
10
11     public void run(){
12         System.out.println(threadName);
13     }
14
15     public static void main(String[] args) throws Excep
16         String tt = args[0];
17         RunnableDemo T1 = new RunnableDemo(tt);
18         Thread t = new Thread(T1);
19         t.start();
20     }
21 }
22
```

« 1 / 95 » test.ql on multi-thread3.testproj - finished in 2 seconds (18810 results) [2024/5/17 18:00:55] Open test.ql

#select 18810 results

#	node	[1]
1	Runnable	ExprNode
2	String	ExprNode
3	String	ExprNode
4	...=...	ExprNode
5	...=...	RelevantNode
6	threadName	ExprNode
7	name	ExprNode
8	name	RelevantNode
9	System.out	ArgNode
10	System.out	ExprNode
11	System.out	RelevantNode
12	System.out	ArgumentNode
13	threadName	ArgNode
14	threadName	ExprNode
15	threadName	RelevantNode
16	threadName	CastingNode
17	threadName	ArgumentNode
18	System	ExprNode
19	...[]	ExprNode
20	String	ExprNode
21	String	ExprNode
22	tt	ExprNode
23	...[...]	ExprNode
24	...[...]	RelevantNode
25	...[...]	CastingNode
26	args	ExprNode
27	args	RelevantNode
28	0	ExprNode
29	0	DefaultXssSanitizer
30	0	DefaultLdapSanitizer
31	0	DefaultMvelInjectionSan
32	RunnableDemo	ExprNode
33	T1	ExprNode
34	new RunnableDemo(...)	NewExpr
35	new RunnableDemo(...)	ExprNode
36	new RunnableDemo(...)	OutNodeExt
37	new RunnableDemo(...)	RelevantNode
38	new RunnableDemo(...)	CastingNode
39	RunnableDemo	ExprNode
40	tt	ArgNode
41	tt	ExprNode
42	tt	RelevantNode
43	tt	ArgumentNode
44	Thread	ExprNode
45	t	ExprNode
46	new Thread(...)	NewExpr
47	new Thread(...)	ExprNode
48	new Thread(...)	OutNodeExt
49	new Thread(...)	RelevantNode
50	new Thread(...)	CastingNode
51	Thread	ExprNode
52	T1	ArgNode

CodeQL DataFlow分析

StageX::fwdFlow

步骤	例子	逻辑
Source		污点分析source 节点
Local Flow	mid = "taint"; node = mid;	程序内分析，如果存在mid 点，能够程序内传播至node，那么认为node 也是流中的一点
Jump Step		为用户提供的自定义 扩展方法，注意无上下文支持
Store	node.filed = mid; node[x] = mid;	为Field、Array、Collection、Map 等赋值
Load	node = mid.filed; node = mid[x];	从Field、Array、Collection、Map 等取值

Call In	public void m(param){
	this.f=param;
	}
	o.m(arg);
		方法调用时候的传播，从实参arg 传播至形参param；注意包括o 到this

Call Out	public void m(param){
	ret=source;
	return ret;
	};
	r=o.m(arg);
		特指return ret 不是从参数传播而来，而是从例如source 点之类的传播而来，这个时候ret 传播至表达式o.m(arg)

Call Through	public void m(param, o){
	ret=param;
	o.field=param;
	return ret;
	}; r=o.m(arg, obj);
		特指return ret 是从参数传播而来，这个时候ret 传播至表达式o.m(arg, obj)。注意return 只是一种类型的ReturnNode，还有一种PostUpdateNode，表示经过Call 之后，其值会变化，比如Callable m 中o 的field已经被改变了，对应传播关系为：实参arg –> 形参param –> o.field –> o<.field>[post update] –> obj<.field>[post update]。

CodeQL DataFlow分析

Stage1::fwdFlow

KCon

t > codeql-home > default > custom > test > query-tests > multi-thread > src > J RunnableDemo.java

```
1  import java.lang.Runnable;
2
3
4  public class RunnableDemo implements Runnable {
5      private String threadName;
6
7      RunnableDemo(String name){
8          threadName = name;
9      }
0
1      public void run(){
2          System.out.println(threadName);
3      }
4
5      public static void main(String[] args) throws Exception {
6          String tt = args[0];
7          RunnableDemo T1 = new RunnableDemo(tt);
8          Thread t = new Thread(T1);
9          t.start();
0      }
1  }
2
3
4
```

« 1 / 1 »

Quick evaluation of DataFlowImpl.qll:1415 on multi-thread.testproj - finished in 3 seconds (16 results) [2024/8/8 22:11:23]

Open DataFlowImpl.qll

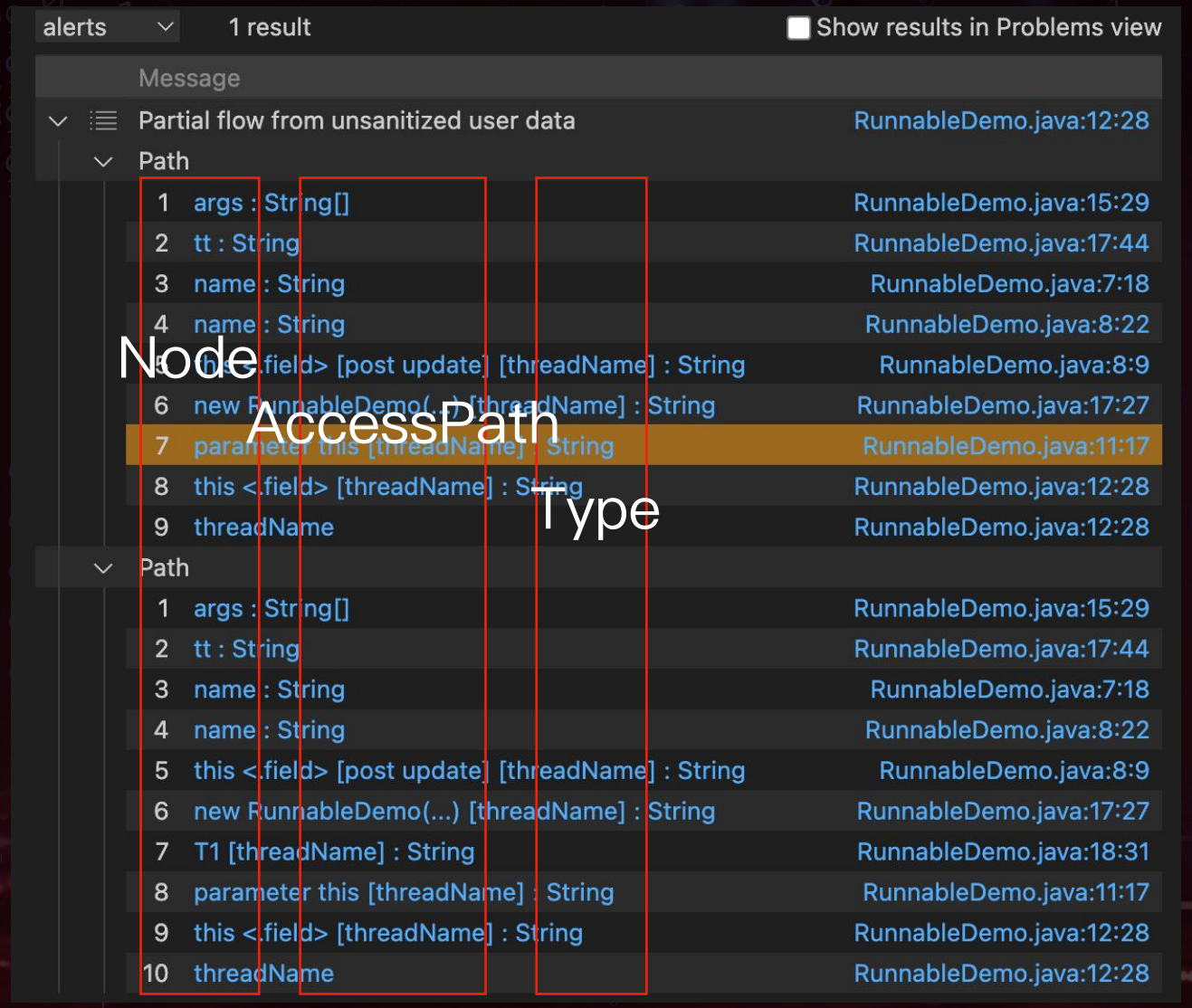
#Quick_evaluation_of_predicate_testFwdFlowWithPre

16 results

#	node	cc	config	pre	step	depth ▲
1	args	false	MyTaintTrackingConfig	args	source	0
2	args	false	MyTaintTrackingConfig	args	localFlowStep	1
3	...[...]	false	MyTaintTrackingConfig	args	localFlowStep	2
4	tt	false	MyTaintTrackingConfig	...[...]	localFlowStep	3
5	name	true	MyTaintTrackingConfig	tt	flow into a callable	4
6	name	true	MyTaintTrackingConfig	name	localFlowStep	5
7	...=...	true	MyTaintTrackingConfig	name	localFlowStep	6
8	this <.field> [post update]	true	MyTaintTrackingConfig	name	store	6
9	new RunnableDemo(...)	false	MyTaintTrackingConfig	this <.field> [post update]	flow through	7
10	T1	false	MyTaintTrackingConfig	new RunnableDemo(...)	localFlowStep	8
11	parameter this	false	MyTaintTrackingConfig	new RunnableDemo(...)	jumpStep	8
12	parameter this	false	MyTaintTrackingConfig	T1	jumpStep	9
13	this <.field>	false	MyTaintTrackingConfig	parameter this	localFlowStep	9
14	threadName	false	MyTaintTrackingConfig	this <.field>	read	10
15	this <.field>	false	MyTaintTrackingConfig	parameter this	localFlowStep	10
16	threadName	false	MyTaintTrackingConfig	this <.field>	read	11

CodeQL DataFlow分析

Node -> PathNode



- Node: 节点
- AccessPath: 路径
- Type: 类型

CodeQL DataFlow分析

DataFlow::AccessPath

API: Predicate Ap apCons(TypeContent tc, AP t

功能：根据TypedContent tc 和上一个节点Node 关联的A

参数：

- TypedContent tc
- Ap tail

返回：

- Ap cons

Stage	AppApprox	Ap	ApNil
Stage1	Unit	Unit	
Stage2	Unit	boolean	false
Stage3	boolean	AccessPathFront	AccessPat
Stage4	AccessPathFront	AccessPathApprox	AccessPat

步骤	Pre Node	ApOption argAp	AP ap
source		apNone()	getApNil(node)
localFlowStep	Node mid	mid.argAp	- 如果非additionalLocalFlowStep，那么为mid.ap - 否则为apNone()
jumpStep	Node mid	apNone()	mid.ap
additionalJumpStep		apNone()	getApNil(node)
store	- Node mid - TypedContent tc	mid.argAp	node.ap = apCons(tc, mid.ap)
read	- Node mid - TypedContent tc	mid.argAp	mid.ap = apCons(tc, node.ap)
flow into	ArgNodeEx arg	- 如果上一个PreStage中，当前ParameterNode node 的AP approx = getApprox(arg.ap)，那么node.argAp = apSome(arg.ap) - 否则node.argAp = apNone()	arg.ap
flow out	ReturnNode ret	ret.argAp	ret.ap
flow through	- ReturnNode ret - ParameterNode p	满足ret.argAp = apSome(p.ap) node.argAp = p.argAp	ret.ap

- argAp：保持和上一个ArgNode 的argAp 相同，主要体现在方法调用的时候
- ap：在store/read 中会变动，否则为上一个节点的取值

CodeQL DataFlow分析

PathNode

KCon

- Node
- AccessPath
- SummaryCtx

步骤	Pre Node	SummaryCtx sc	AP ap
source		SummaryCtxNone	TAccessPathNil(node.getDataFlowType())
localFlowStep	PathNodeMid mid	mid.sc	- 如果非additionalLocalFlowStep，那么为mid.ap - 否则为apNone()
jumpStep	PathNodeMid mid	SummaryCtxNone	mid.ap
additionalJumpStep & additionalJumpStateStep	PathNodeMid mid	SummaryCtxNone	TAccessPathNil(node.getDataFlowType())
store	- PathNodeMid mid - TypedContent tc	mid.sc	mid.ap=node.ap.pop(tc)
read	- PathNodeMid mid - TypedContent tc	mid.sc	mid.ap = node.ap.push(tc)
flow into	PathNodeMid mid	- TSummaryCtxSome(p, state, ap) - 或者不存在以上时，TSummaryCtxNone()	mid.ap
flow out	ReturnNode ret	SummaryCtxNone	mid.ap
flow through	- ReturnNode ret - ArgumentNode mid	mid.sc	ret.ap

PART ONE

03

解决CodeQL Java 代码分析难点

CodeQL Java Optimisation

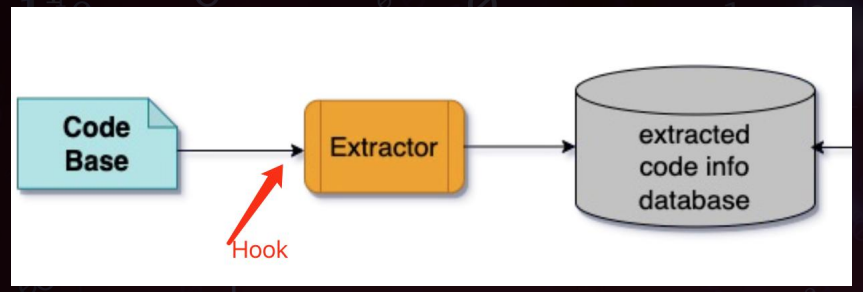
Runnable Interface

```
import java.lang.Runnable;

public class RunnableDemo implements Runnable {
    private String threadName;

    RunnableDemo(String name){
        threadName = name;
    }

    public void run(){
        System.out.println(threadName);
    }

    public static void main(String[] args) throws Exception {
        String tt = args[0];
        RunnableDemo T1 = new RunnableDemo(tt);
        Thread t = new Thread(T1);
        t.start();
    }
}
```

- 方法 1：Patch 源码
- 方法 2：Patch CodeQL DB
- 方法 3：Patch CodeQL DFA

CodeQL Java Optimisation

Adapting Java Runnable Interface Option 1: Patch Source Code




```java
public static void main(String[] args) throws Exception {
    String tt = args[0];
    RunnableDemo T1 = new RunnableDemo(tt);
    Thread t = new Thread(T1);
    t.start();
    // patch
    t.run();
}
```

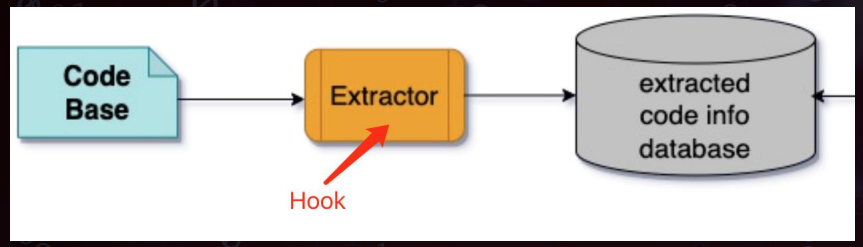
1. Hook com.semmle.extractor.java.JavaExtractor#main
2. Use javaparser ModifierVisitor Dynamic Patch Source Code

```java
public class JavaExtractor {

    public static void main(String[] args) {
        String allArgs = StringUtil.glue( separator: " ", args);
        JavaExtractor extractor = new JavaExtractor(args);
        boolean hasJavacErrors = false;

        extractor.patch();
```

```java
    private static class JavaThreadModifier extends ModifierVisitor<List<String>> {
        @Override
                                                        增加在当前stmts后
                                                        加在block 最后面
                                                    odCallExpr n, List<String> arg) {
                                                    lve().getQualifiedSignature();
                System.out.println("[MethodCallExpr] signature found: " + signature);
                System.out.println(n);
                switch (signature) {
                    case "java.lang.Thread.start()":
                        n.findAncestor(BlockStmt.class).ifPresent(blockStmt -> {
                            // blockStmt.getStatement(blockStmt.getStatements().indexOf(n) + 1).setLi
                            // blockStmt.addOrphanComment(new LineComment("auto add java.lang.Thread#
                            // support LambdaExpr
                            if (n.getChildNodes().size() == 2){
                                blockStmt.addStatement(n.getScope().get().toString() + ".run();");
                                arg.add(n.toString());
                            }
                        });
                        break;
                    case "java.util.concurrent.Executor.execute(java.lang.Runnable)":
                    case "java.util.concurrent.ExecutorService.submit(java.lang.Runnable)":
                        n.findAncestor(BlockStmt.class).ifPresent(blockStmt -> {
                            // support LambdaExpr
                            if (n.getChildNodes().size() == 2){
                                blockStmt.addStatement(n.getScope().get().toString() + ".run();");
                                arg.add(n.toString());
                            }
                        });
```

CodeQL Java Optimisation

Adapting Java Runnable Interface Option 2: Patch DB Trap File



Hook com.semmle.extractor.java.ClassDeclExtractor#visitExec

1. Stmts
2. Expr
3. Subexpr
4. Callable

1. Stmt: R1.start();
#10092=*
stmts(#10092,14,#10035,3,#10033)
#10093=*
locations_default(#10093,#10000,9,9,9,19)
hasLocation(#10092,#10093)

2. Expr: R1.start();
#10114=*
exprs(#10114,61,#10014,#10092,0)
callableEnclosingExpr(#10114,#10033)
statementEnclosingExpr(#10114,#10092)
#10115=*
locations_default(#10115,#10000,9,9,9,18)
hasLocation(#10114,#10115)

3. subexpr: R1
#10116=*
exprs(#10116,60,#10079,#10114,–1)
callableEnclosingExpr(#10116,#10033)
statementEnclosingExpr(#10116,#10092)
#10117=*
locations_default(#10117,#10000,9,9,9,10)
hasLocation(#10116,#10117)

4. callable
#10118=@"callable;{#10079}.start(){#10014}"
callableBinding(#10114,#10118)
variableBinding(#10116,#10083)

1. Stmt: R1.run();
#10094=*
stmts(#10094,14,#10035,4,#10033)
#10095=*
locations_default(#10095,#10000,10,9,10,17)
hasLocation(#10094,#10095)

2. expr: R1.run();
#10119=*
exprs(#10119,61,#10014,#10092,0)
callableEnclosingExpr(#10119,#10033)
statementEnclosingExpr(#10119,#10092)
#10120=*
locations_default(#10120,#10000,10,9,10,16)
hasLocation(#10119,#10120)

3. subexpr: R1
#10121=*
exprs(#10121,60,#10079,#10119,–1)
callableEnclosingExpr(#10121,#10033)
statementEnclosingExpr(#10121,#10092)
#10122=*
locations_default(#10122,#10000,10,9,10,10)
hasLocation(#10121,#10122)

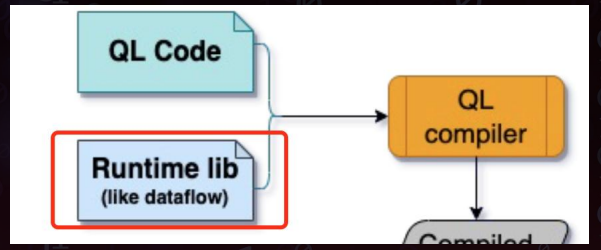
4. callable
#10123=@"callable;{#10079}.run(){#10014}"
callableBinding(#10119,#10123)
variableBinding(#10121,#10083)

CodeQL Java Optimisation

Adapting Java Runnable Interface Option 3: Patch DFA

KCon

- JumpStep API: AdditionalValueStep




```
/**
 * Holds if data can flow from `node1` to `node2` through a field or
 * variable capture.
 */
predicate jumpStep(Node node1, Node node2) {
  fieldStep(node1, node2)
  or
  any(AdditionalValueStep a).step(node1, node2) and
  node1.getEnclosingCallable() != node2.getEnclosingCallable()
  or
  FlowSummaryImpl::Private::Steps::summaryJumpStep(node1.(FlowSummaryNode).getSummaryNode(),
    node2.(FlowSummaryNode).getSummaryNode())
}
```

CodeQL Java Optimisation

Adapting Java Runnable Interface Option 3: Patch DFA

KCon

```
/** Value step from the constructor call of a `Runnable` to the instance parameter (this) of `run`. */
private class RunnableStartToRunStep extends AdditionalValueStep {
  override predicate step(Node pred, Node succ) {
    exists(ConstructorCall cc, Method m |
      m.getDeclaringType() = cc.getConstructedType().getSourceDeclaration() and
      cc.getConstructedType().getAnAncestor().hasQualifiedName("java.lang", "Runnable") and
      m.hasName("run")
    |
      pred.asExpr() = cc and
      succ.(InstanceParameterNode).getEnclosingCallable() = m
    )
  }
}
```

- 放在DataFlowImpl.qll，或者更底层的依赖，例如DataFlowPrivate.qll

CodeQL Java Optimisation

Adapting Java Runnable Interface Option 3: Patch DFA

KCon

```java
import java.lang.Runnable;


public class RunnableDemo implements Runnable {
    private String threadName;

    RunnableDemo(String name){
        threadName = name;
    }

    public void run(){
        System.out.println(threadName);
    }

    public static void main(String[] args) throws Exception {
        String tt = args[0];
        RunnableDemo T1 = new RunnableDemo(tt);
        Thread t = new Thread(T1);
        t.start();
    }
}
```

« 1 / 1 »
in 2 seconds (13 results) [2024/5/16 20:12:42]
DataFlowImpl.qll

#Quick_evaluation_of_predicate_testFwdFlow
13 results

#	node	cc	config	step
1	...=...	true	MyTaintTrackingConfig	localFlowStep
2	name	true	MyTaintTrackingConfig	localFlowStep
3	threadName	false	MyTaintTrackingConfig	read
4	...[...]	false	MyTaintTrackingConfig	localFlowStep
5	args	false	MyTaintTrackingConfig	localFlowStep
6	new RunnableDemo(...)	false	MyTaintTrackingConfig	flow out of a callable
7	tt	false	MyTaintTrackingConfig	localFlowStep
8	T1	false	MyTaintTrackingConfig	localFlowStep
9	this <.field> [post update]	true	MyTaintTrackingConfig	store
10	this <.field>	false	MyTaintTrackingConfig	localFlowStep
11	args	false	MyTaintTrackingConfig	source
12	name	true	MyTaintTrackingConfig	flow into a callable
13	parameter this	false	MyTaintTrackingConfig	jumpStep

CodeQL Java Optimisation

多线程进阶

KCon

```java
import java.lang.Thread;

public class ThreadDemo extends Thread {
    private String threadName;

    ThreadDemo() {}

    public void setThreadName(String name){threadName=name;}

    public void run() {
        System.out.println(threadName);
    }

    public static void main(String[] args) throws Exception {
        String tt = args[0];
        ThreadDemo T1 = new ThreadDemo();
        T1.setThreadName(tt);
        T1.start();
    }
}
```

如何应对?

```java
import java.lang.Thread;

public class ThreadDemoWhile extends Thread {
    private String threadName;

    ThreadDemoWhile() {}

    public void setThreadName(String name){threadName=name;}

    public void run(){
        while(true){
            System.out.println(threadName);
        }
    }

    public static void main(String[] args) throws Exception {
        String tt = args[0];
        ThreadDemoWhile T1 = new ThreadDemoWhile();
        T1.start();
        T1.setThreadName(tt);
    }
}
```

阁下又该如何应对?

CodeQL Java Optimisation

Debug DFA

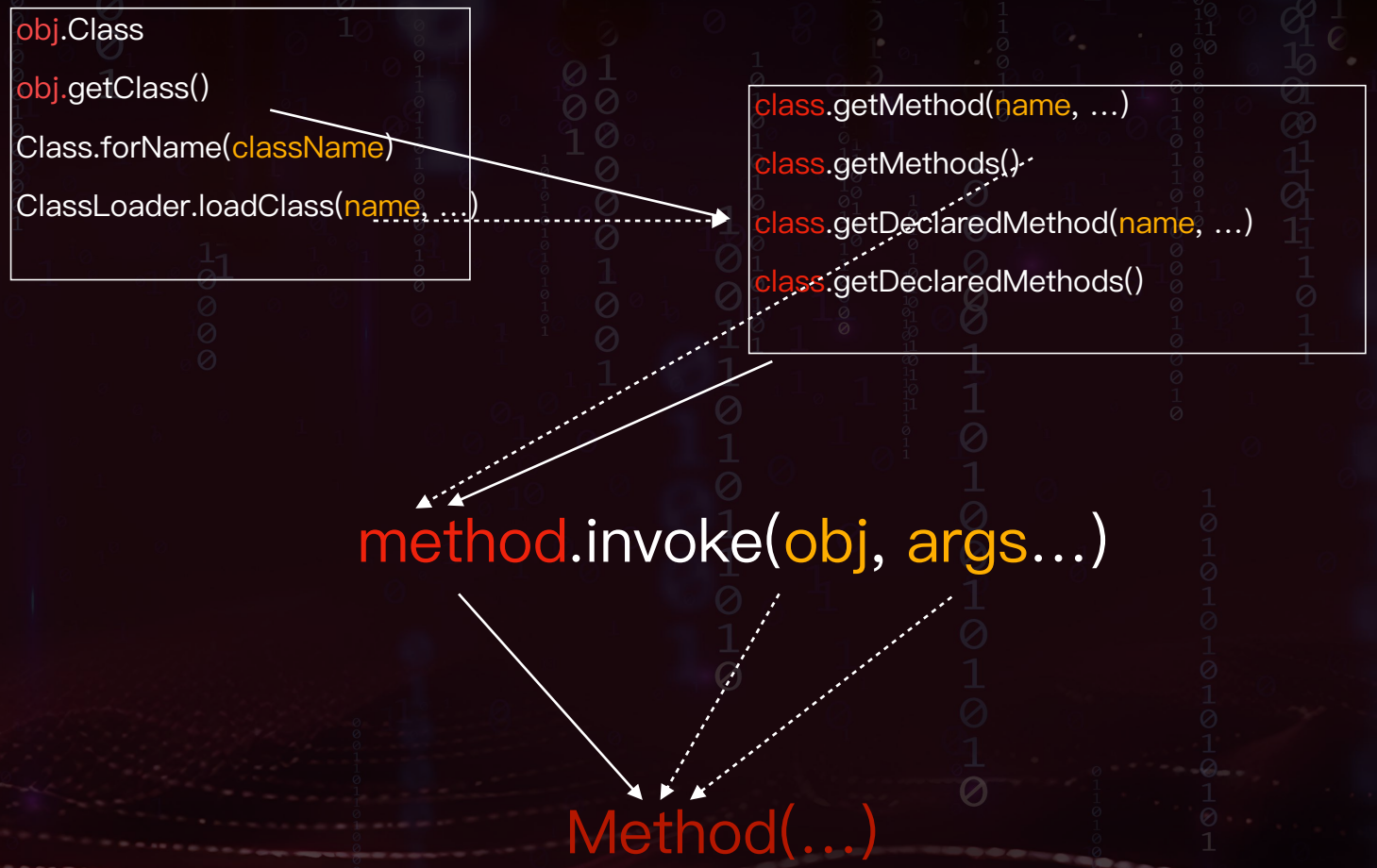
- 在DataFlowImpl 中加入一个DataFlow::Configuration 实现类
- isAdditionalFlowStep 实现了args -> args[0]
- 在DataFlowImpl 中通过“Quick Evaluation” 实现Debug

```
class MyTaintTrackingConfig extends Configuration {
  MyTaintTrackingConfig() { this = "MyTaintTrackingConfig" }

  override predicate isSource(Node source) {
    exists(Method m |
      m.hasName("main")
      and m.getAParameter() = source.asParameter()
    )
  }

  override predicate isSink(Node sink) {
    exists(MethodAccess ma |
      ma.getMethod().hasName("println") and
      sink.asExpr() = ma.getAnArgument()
    )
  }

  override predicate isAdditionalFlowStep(Node src, Node sink) {
    // defaultAdditionalTaintStep(node1, node2)
    // 来自TaintTrackingUtil，但是不能直接import，存在依赖关系
    exists(Content f |
      readStep(src, f, sink) and
      not sink.getTypeBound() instanceof PrimitiveType and
      not sink.getTypeBound() instanceof BoxedType and
      not sink.getTypeBound() instanceof NumberType and
      (
        containerContent(f)
        or
        f instanceof TaintInheritingContent
      )
    )
  }
}
```

CodeQL Java Optimisation

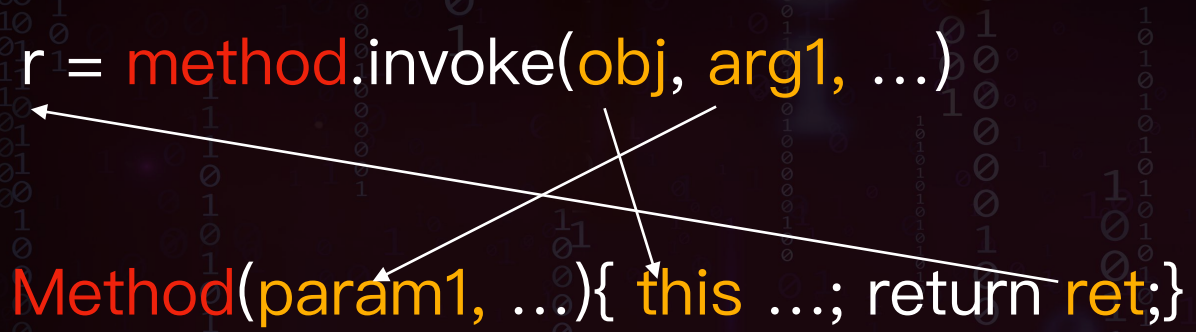
Adapting Java Reflection Step 1: Reflection Analysis

KCon



CodeQL Java Optimisation

Adapting Java Reflection Step 2: Inclusion reflection analyses result into DFA



- Call In
  - arg 传播至parma
  - obj 传播至Method 的this
- Call Through
  - RetNode 要传播至r
  - Method 里如果有PostUpdateNode，也要传播至对应的obj/arg

一种Patch DataFlowImpl 的方案：

- Stage1
  - fwdFlow
    - viableParamArgEx
    - fwdFlowOut
    - fwdFlowIsEntered
  - revFlow
    - viableReturnPosOutEx
- pathStep
  - pathIntoCallable
  - pathThroughCallable
- Subpaths
  - subpaths02

PART ONE

04

历史漏洞回溯

历史漏洞回溯

ActiveMQ CVE-2023-46604 RCE

```java
public abstract class BaseDataStreamMarshaller implements DataStreamMarshaller {

    private Throwable createThrowable(String className, String message) {
        try {
            Class clazz = Class.forName(className, false, BaseDataStreamMarshaller.class.getClassLoader());
            OpenWireUtil.validateIsThrowable(clazz);
            Constructor constructor = clazz.getConstructor(new Class[] {String.class});
            return (Throwable)constructor.newInstance(new Object[] {message});
        } catch (IllegalArgumentException e) {
            return e;
        } catch (Throwable e) {
            return new Throwable(className + ": " + message);
        }
```

1. source -> start

socket.getInputStream
TcpTransport#initializeStreams
TcpTransport#connect
TcpTransport#doStart
TransportThreadSupport#doStart

2. run -> sink

TcpTransport#run
TcpTransport#doRun
TcpTransport#readCommand
OpenWireFormat#unmarshal
OpenWireFormat#doUnmarshal
ExceptionResponseMarshaller#tightUnmarshal
ConnectionErrorMarshaller#tightUnmarshal
MessageAckMarshaller#tightUnmarshal
BaseDataStreamMarshaller#looseUnmarsalThrowable
BaseDataStreamMarshaller#tightUnmarsalThrowable
BaseDataStreamMarshaller#createThrowable

历史漏洞回溯

ActiveMQ CVE-2023-46604 RCE

getInputStream(...) : InputStream	TcpTransport.java
new DataInputStream(...) : DataInputStream	TcpTransport.java
this [post update] : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this <.method> [post update] : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this <.method> [post update] : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
super : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
parameter this : TcpTransport [dataIn] : DataInputStream	TransportThreadSupport.java
this : TcpTransport [dataIn] : DataInputStream	TransportThreadSupport.java
parameter this : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this <.method> : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
parameter this : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this <.method> : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
parameter this : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this <.field> : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
dataIn : DataInputStream	TcpTransport.java
dis : DataInputStream	OpenWireFormat.java
dataIn : DataInputStream	OpenWireFormat.java
dis : DataInputStream	OpenWireFormat.java

Close

File: activemq-client-jakarta/target/generated-sources/apache/activemq/transport/TransportThreadSupport.java

```
26
27      private boolean daemon;
28      private Thread runner;
29      // should be a multiple of 128k
30      private long stackSize;
31
32      public boolean isDaemon() {
33          return daemon;
34      }
35
36      public void setDaemon(boolean daemon) {
37          this.daemon = daemon;
38      }
39
40      protected void doStart() throws Exception {
41          runner = new Thread(null, this, "ActiveMQ Transport: " + toString(), stackSize);
42          runner.setDaemon(daemon);
43          runner.start();
44      }
45
46      /**
47       * @return the stackSize
48       */
49      public long getStackSize() {
50          return this.stackSize;
51      }
52
53      /**
54       * @param stackSize the stackSize to set
55       */
56      public void setStackSize(long stackSize) {
57          this.stackSize = stackSize;
58      }
59  }
60
```

历史漏洞回溯

ActiveMQ CVE-2023-46604 RCE

new DataInputStream(...) : DataInputStream	TcpTransport.java
this [post update] : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this <.method> [post update] : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this <.method> [post update] : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
super : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
parameter this : TcpTransport [dataIn] : DataInputStream	TransportThreadSupport.java
this : TcpTransport [dataIn] : DataInputStream	TransportThreadSupport.java
parameter this : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this <.method> : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
parameter this : TcpTransport [dataIn] : DataInputStream	TcpTransport.java
this <.method> : TcpTransport [dataIn] : DataInputStream	TcpTransport.java

Close

File: activemq-client-jakarta/target/generated-sources/apache/activemq/transport/tcp/TcpTransport.java

```
201     public String toString() {
202         return "" + (socket.isConnected() ? "tcp://" + socket.getInetAddress() + ":" + soc
203                 : (localLocation != null ? localLocation : remoteLocation)) ;
204     }
205
206     /**
207      * reads packets from a Socket
208      */
209     @Override
210     public void run() {
211         LOG.trace("TCP consumer thread for " + this + " starting");
212         this.runnerThread=Thread.currentThread();
213         try {
214             while (!isStopped() && !isStopping()) {
215                 doRun();
216             }
217         } catch (IOException e) {
218             stoppedLatch.get().countDown();
219             onException(e);
220         } catch (Throwable e){
221             stoppedLatch.get().countDown();
222             IOException ioe=new IOException("Unexpected error occurred: " + e);
223             ioe.initCause(e);
224             onException(ioe);
225         }finally {
226             stoppedLatch.get().countDown();
227         }
228     }
229
230     protected void doRun() throws IOException {
231         try {
```

历史漏洞回溯

RocketMQ CVE-2023-33246 RCE

KCon

挑战1: 跨3个线程

1. Netty server 到NettyRequestProcessor processRequest 处理请求
2. AdminBrokerProcessor 线程，更新brokerController.brokerConfig，其中有个属性rocketmqHome
3. 周期线程FilterServerManager，会用rocketmqHome 拼接成命令执行，触发漏洞

挑战2: MixAll#properties2Object Invoke

```
public static void properties2Object(final Properties p, final Object object) {
    Method[] methods = object.getClass().getMethods();
    for (Method method : methods) {
        String mn = method.getName();
        if (mn.startsWith("set")) {
            try {
                String tmp = mn.substring(4);
                String first = mn.substring(3, 4);
                String key = first.toLowerCase() + tmp;
                String property = p.getProperty(key);
                if (property != null) {
                    Class<?>[] pt = method.getParameterTypes();
                    if (pt != null && pt.length > 0) {
                        String cn = pt[0].getSimpleName();
                        Object arg = null;
                        if (cn.equals("int") || cn.equals("Integer")) {
                            ...
                        method.invoke(object, arg);
```

挑战3: 内部类调用外部类方法

```
public class FilterServerManager {
    ...
    public void start() {

        this.scheduledExecutorService.scheduleAtFixedRate(new
AbstractBrokerRunnable(brokerController.getBrokerConfig()) {
            @Override
            public void run0() {
                try {
                    FilterServerManager.this.createFilterServer();
                } catch (Exception e) {
                    log.error("", e);
                }
            }
        }, 1000 * 5, 1000 * 30, TimeUnit.MILLISECONDS);
    }

    public void createFilterServer() {
```

历史漏洞回溯

RocketMQ CVE-2023-33246 RCE

KCon

Users > m0d9 > study > codeql-home > default > db > rqdb_v2.13.5 > src.zip > home > m0d1 > Downloads > rocketmq-roc

```
55  public class MixAll {
347     public static void properties2Object(final Properties p, final Object object) {
349         for (Method method : methods) {
383             }
352                 try {
356                     String key = first.toLowerCase() + tmp;
357                     String property = p.getProperty(key);
358                     if (property != null) {
359                         Class<?>[] pt = method.getParameterTypes();
360                         if (pt != null && pt.length > 0) {
361                             String cn = pt[0].getSimpleName();
362                             Object arg = null;
363                             if (cn.equals("int") || cn.equals("Integer")) {
364                                 arg = Integer.parseInt(property);
365                             } else if (cn.equals("long") || cn.equals("Long")) {
366                                 arg = Long.parseLong(property);
367                             } else if (cn.equals("double") || cn.equals("Double")) {
368                                 arg = Double.parseDouble(property);
369                             } else if (cn.equals("boolean") || cn.equals("Boolean")) {
370                                 arg = Boolean.parseBoolean(property);
371                             } else if (cn.equals("float") || cn.equals("Float")) {
372                                 arg = Float.parseFloat(property);
373                             } else if (cn.equals("String")) {
374                                 arg = property;
375                             } else {
376                                 continue;
377                             }
378                             method.invoke(object, arg);
379                         }
380                     }
381                 } catch (Throwable ignored) {
382                 }
383             }
384         }
385     }
```

« 1 / 1 » CVE-2023-33246 Open command-exec.ql

20	str : String	MixAll.java:301:48
21	str : String	MixAll.java:304:55
22	getBytes(...) : byte[]	MixAll.java:304:55
23	in : ByteArrayInputStream	MixAll.java:305:29
24	properties [post update] : Properties	MixAll.java:305:13
25	properties : Properties	MixAll.java:311:16
26	string2Properties(...) : Properties	BrokerContainerProcessor.java:97:36
27	brokerProperties : Properties	BrokerContainerProcessor.java:110:34
28	p : Properties	MixAll.java:347:42
29	p : Properties	MixAll.java:357:39
30	getProperty(...) : Object	MixAll.java:357:39
31	arg : Number	MixAll.java:378:51
32	object [post update] : BrokerConfig [rocketmqHome] : Object	MixAll.java:378:43
33	brokerConfig [post update] : BrokerConfig [rocketmqHome] : Object	BrokerContainerProcessor.java:110:52
34	brokerConfig : BrokerConfig [rocketmqHome] : Object	BrokerContainerProcessor.java:150:63
35	brokerConfig : BrokerConfig [rocketmqHome] : Object	BrokerContainer.java:272:44
36	brokerConfig : BrokerConfig [rocketmqHome] : Object	BrokerContainer.java:275:42
37	brokerConfig : BrokerConfig [rocketmqHome] : Object	BrokerContainer.java:288:51
38	brokerConfig : BrokerConfig [rocketmqHome] : Object	BrokerContainer.java:294:82
39	brokerConfig : BrokerConfig [rocketmqHome] : Object	InnerBrokerController.java:36:9
40	brokerConfig : BrokerConfig [rocketmqHome] : Object	InnerBrokerController.java:39:15
41	brokerConfig : BrokerConfig [rocketmqHome] : Object	BrokerController.java:284:9
42	brokerConfig : BrokerConfig [rocketmqHome] : Object	BrokerController.java:287:14
43	brokerConfig : BrokerConfig [rocketmqHome] : Object	BrokerController.java:291:9
44	brokerConfig : BrokerConfig [rocketmqHome] : Object	BrokerController.java:296:29
45	this [post update] : BrokerController [brokerConfig, rocketmqHome] : Object	BrokerController.java:296:9
46	this : BrokerController [brokerConfig, rocketmqHome] : Object	BrokerController.java:297:9

历史漏洞回溯

RocketMQ CVE-2023-33246 RCE

KCon

```java
public class FilterServerManager {

    public static final long FILTER_SERVER_MAX_IDLE_TIME_MILLS = 30000;
    private static final Logger log = LoggerFactory.getLogger(LoggerName.BROKER_LOGGER_NAME);
    private final ConcurrentMap<Channel, FilterServerInfo> filterServerTable =
        new ConcurrentHashMap<>(16);
    private final BrokerController brokerController;

    private ScheduledExecutorService scheduledExecutorService = Executors
        .newSingleThreadScheduledExecutor(new ThreadFactoryImpl("FilterServerManagerScheduledThrea

    public FilterServerManager(final BrokerController brokerController) {
        this.brokerController = brokerController;
    }

    public void start() {

        this.scheduledExecutorService.scheduleAtFixedRate(new AbstractBrokerRunnable(brokerContro
            @Override
            public void run0() {
                try {
                    FilterServerManager.this.createFilterServer();
                } catch (Exception e) {
                    log.error("", e);
                }
            }
        }, 1000 * 5, 1000 * 30, TimeUnit.MILLISECONDS);
    }

    public void createFilterServer() {
        int more =
            this.brokerController.getBrokerConfig().getFilterServerNums() - this.filterServerTable
        String cmd = this.buildStartCommand();
        for (int i = 0; i < more; i++) {
            FilterServerUtil.callShell(cmd, log);
        }
    }
```

```
rocketmqHome] : Object
84  new InnerBrokerController(...) : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  BrokerContainer.java:294:50
85  brokerController : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  BrokerContainer.java:313:20
86  addDLedgerBroker(...) : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  BrokerContainer.java:275:20
87  addBroker(...) : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  BrokerContainerProcessor.java:150:32
88  brokerController : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  BrokerContainerProcessor.java:163:17
89  parameter this : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  InnerBrokerController.java:56:17
90  this : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  InnerBrokerController.java:57:9
91  this <.method> : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  InnerBrokerController.java:63:9
92  parameter this : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  BrokerController.java:1461:20
93  this : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  BrokerController.java:1492:9
94  this : BrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : Object  BrokerController.java:1527:13
95  this.filterServerManager : FilterServerManager [brokerController, brokerConfig, rocketmqHome] : Object  BrokerController.java:1527:13
96  parameter this : FilterServerManager [brokerController, brokerConfig, rocketmqHome] : Object  FilterServerManager.java:55:17
97  parameter this : FilterServerManager [brokerController, brokerConfig, rocketmqHome] : Object  FilterServerManager.java:69:17
98  this : FilterServerManager [brokerController, brokerConfig, rocketmqHome] : Object  FilterServerManager.java:72:22
99  parameter this : FilterServerManager [brokerController, brokerConfig, rocketmqHome] : Object  FilterServerManager.java:78:20
100 this : FilterServerManager [brokerController, brokerConfig,
```

历史漏洞回溯

RocketMQ CVE-2023-37582 新发现

- BrokerContainer 方式启动的Broker 可绕过属性过滤补丁，5.2.0 后已修复

brokerProperties : Properties	BrokerContainerProcessor.java
p : Properties	MixAll.java
p : Properties	MixAll.java
getProperty(...) : String	MixAll.java
arg : String	MixAll.java
object [post update] : BrokerConfig [rocketmqHome] : String	MixAll.java
brokerConfig [post update] : BrokerConfig [rocketmqHome] : String	BrokerContainerProcessor.java
brokerConfig : BrokerConfig [rocketmqHome] : String	BrokerContainerProcessor.java
brokerConfig : BrokerConfig [rocketmqHome] : String	BrokerContainer.java
brokerConfig : BrokerConfig [rocketmqHome] : String	BrokerContainer.java
brokerConfig : BrokerConfig [rocketmqHome] : String	BrokerContainer.java
brokerConfig : BrokerConfig [rocketmqHome] : String	BrokerContainer.java
brokerConfig : BrokerConfig [rocketmqHome] : String	InnerBrokerController.java
brokerConfig : BrokerConfig [rocketmqHome] : String	InnerBrokerController.java
this <constr(this)> [post update] : InnerBrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : String	InnerBrokerController.java
new InnerBrokerController(...) : InnerBrokerController [filterServerManager, brokerController, brokerConfig, rocketmqHome] : String	BrokerContainer.java

TONGDAO

KCon 2024

THANKS

演讲人：m0d9@Tencent YUNDING LAB

时间：2024.08.24