Smtm模块

smtm模块总的来说应该如下表所示：

smtm模块总的来说应该如下表所示：

Statement syntax	CodeQL class	Superclasses	Remarks
`;`	EmptyStmt		
Expr `;`	ExprStmt		
`{` Stmt `... }`	Block		
`if (` Expr `)` Stmt `else` Stmt	IfStmt	ConditionalStmt	
`if (` Expr `)` Stmt			
`while (` Expr `)` Stmt	WhileStmt	ConditionalStmt, LoopStmt	
`do` Stmt `while (` Expr `)`	DoStmt	ConditionalStmt, LoopStmt	
`for (` Expr `;` Expr `;` Expr `)` Stmt	ForStmt	ConditionalStmt, LoopStmt	
`for (` VarAccess `:` Expr `)` Stmt	EnhancedForStmt	LoopStmt	

Statement syntax	CodeQL class	Superclasses	Remarks
`switch (` Expr `)` `{` SwitchCase `... }`	SwitchStmt		
`try {` Stmt `... }` `finally {` Stmt `... }`	TryStmt		
`return` Expr `;`	ReturnStmt		
`return ;`			
`throw` Expr `;`	ThrowStmt		
`break ;`	BreakStmt	JumpStmt	
`break label ;`			
`continue ;`	ContinueStmt	JumpStmt	
`continue label ;`			
`label :` Stmt	LabeledStmt		
`synchronized (` Expr `)` Stmt	SynchronizedStmt		
`assert` Expr `:` Expr `;`	AssertStmt		
`assert` Expr `;`			
TypeAccess `name ;`	LocalVariableDeclStmt		
`class name {` Member `... } ;`	LocalClassDeclStmt		
`this (` Expr `, ... ) ;`	ThisConstructorInvocationStmt		
`super (` Expr `, ... ) ;`	SuperConstructorInvocationStmt		
`catch (` TypeAccess `name )` `{` Stmt `... }`	CatchClause		can only occur as child of a TryStmt

Statement syntax	CodeQL class	Superclasses	Remarks
`case` Literal `:` Stmt `...`	ConstCase		can only occur as child of a SwitchStmt
`default :` Stmt `...`	DefaultCase		can only occur as child of a SwitchStmt

通常情况下我们会在addtionalTaintStep中定义和描述一些情况来帮助判断数据流：
例如：

```
class ExceptionTaintStep extends TaintTracking::AdditionalTaintStep {
  override predicate step(DataFlow::Node n1, DataFlow::Node n2) {
    exists(Call call, TryStmt try, CatchClause catch, MethodAccess getMessageCall |
      // the call is within the `try` block, which has a corresponding `catch` clause
      call.getEnclosingStmt().getEnclosingStmt*() = try.getBlock() and
      try.getACatchClause() = catch and
      // the `catch` clause is likely to catch an exception thrown by the call
      (
        catch.getACaughtType().getASupertype*() = call.getCallee().getAThrownExceptionType() or
        catch.getACaughtType().getASupertype*() instanceof TypeRuntimeException
      ) and
      // the exception message is read by `getMessageCall` within the `catch` block
      catch.getVariable().getAnAccess() = getMessageCall.getQualifier() and
      getMessageCall.getMethod().getName().regexpMatch("get(Localized)?Message|toString") and
      // taint flows from any argument of the call to a place where the exception message is accessed
      n1.asExpr() = call.getAnArgument() and
      n2.asExpr() = getMessageCall
    )
  }
}
```

这里的代码片段是递归检测try-catch分支的情况。在一些情况下，因为没有定义这些node间的关系，以至于我们的逻辑走到exception就丢失，导致扫描无法成功。