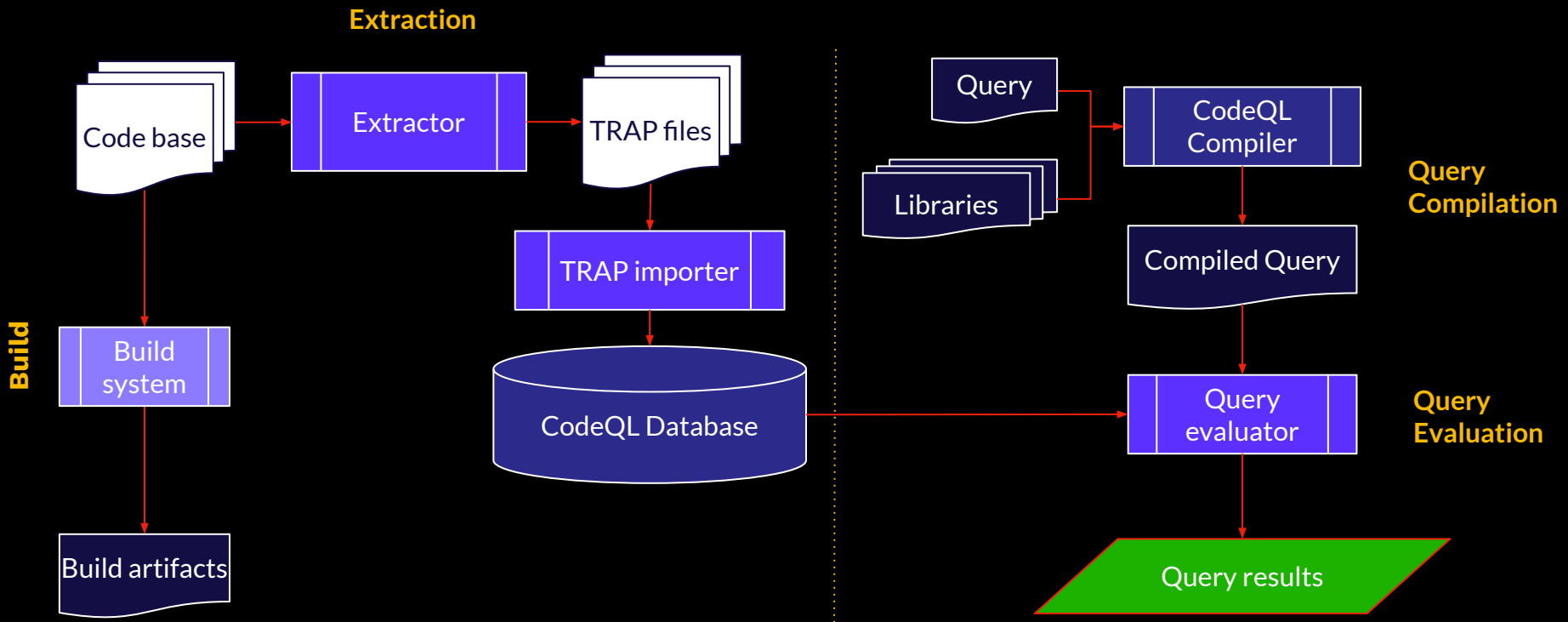
GitHub

CodeQL Query Writing

Michael Hohn, hohn@github.com

Overview the CodeQL system

The Big Picture: Databases and Queries



Overview of the CodeQL language and tools

CodeQL is...

- a **logic language** based on first-order logic
- a **declarative language** without side effects
- an **object-oriented language**
- a **query language** working on a read-only snapshot database

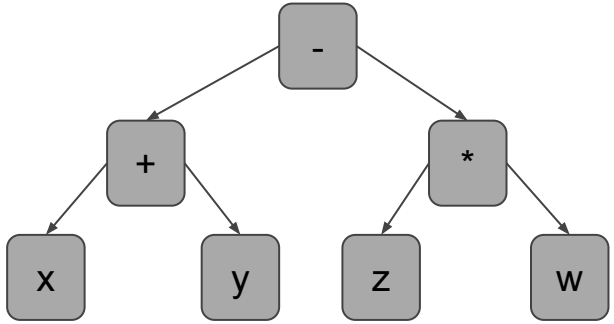
+

- rich **standard libraries** for program analysis
- **tools to create databases** from source code
- CLI and IDE extensions

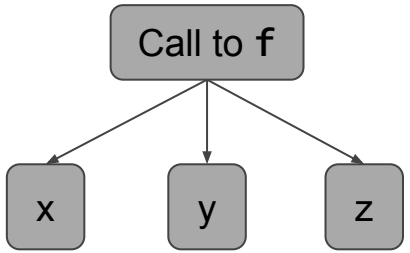
Representing a program: Abstract Syntax Trees

- Abstract syntax trees have a node for each program element
- Hide some of the complexity of parsing
- Starting point for most program analysis

```
x + y - z * w
```



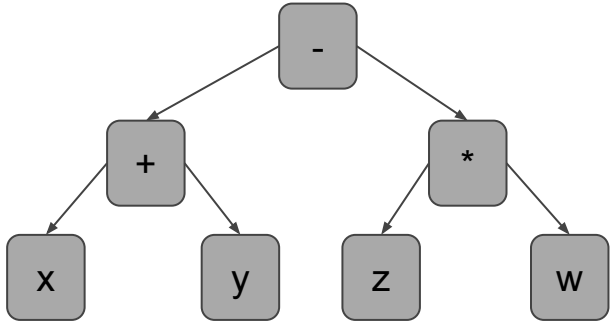

```
f(x, y, z)
```



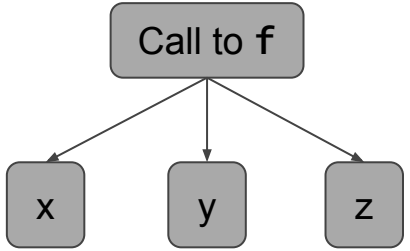
Representing a program: QL class hierarchy

- QL classes for each kind of expression, statement, etc.
- Methods to get particular child nodes

`x + y - z * w`



`f(x, y, z)`



Overview of data flow and taint tracking

Data Flow and Taint Tracking

Basic question - what sources of untrusted information can influence these values that are used in dangerous way

Shows up frequently in security queries

- XSS
- SQL/Code/Path injections
- Encryption issues

Data flow analysis

- Model the program as a directed graph
- Nodes are program elements that have values
- Edges represent steps that copy data from one node to another

Data flow analysis

Within a function, we can compute every path that data can take, but this isn’t feasible for a whole program

We solve this by constraining the sources and sinks before evaluating the flow