Detect complex code patterns using semantic grep

Drew Dennison | r2c.dev
@r2cdev @drewdennison
whois?

**me:**
Drew Dennison, co-founder @ r2c
BS in CompSci from MIT, ex-Palantir dev

**r2c**
We’re an SF based static analysis startup on a mission to profoundly improve software security and reliability.
tl;dr

- Computers excel at **consistently** applying **human** experience
- The coding future has a “paved road” of default-safe frameworks (i.e. React or ORMs)
- `grep` and static analysis tools can be slow, complicated, or noisy
- We need a tool that is fast, easy to use, precise, multi-lingual, and open source!

**Semgrep**: Lightweight static analysis for many languages. Find bug variants with patterns that look like source code.
Outline

1. **Demo**: Dive right in!
2. **Tech Talk**: `grep` and Abstract Syntax Trees (ASTs)
3. **In Practice**: Your rules and community rule packs
import flask
from flask import response as r

app = Flask()

@app.route("/index")
def index():
    rep = r.set_cookie("hello", "world")
    return rep

@app.route("/snafu")
def index():
    rep = r.set_cookie("hello", generate_cookie_value("world"), secure=False)
    return rep

# @app.route("/snafu")
# def index():
#    rep = r.set_cookie("hello", "world", secure=False)
#    return rep

@app.route("/admin")
def admin():
Node Exec

```javascript
exec("ls");
```

⇒ [https://semgrep.live/Xnw](https://semgrep.live/Xnw)

Full Solution: [https://semgrep.live/1Kk](https://semgrep.live/1Kk)
Finding Uses of `unsafe`

```go
unsafe.Pointer(intPtr)
unsafe.Sizeof(intArray[0])
```

⇒ https://semgrep.live/nJNZ

Full Solution: https://semgrep.live/ZqLp
Injection Sending File

```python
@app.route("/get_file/<filename>")
def get_file(filename):
    print("sending file", filename)
    return send_file(filename, as_attachment=True)
```

⇒ https://semgrep.live/4bXx

Full Solution: https://semgrep.live/Pevp
@app.route("/index")
def index():
    r = response.set_cookie("username","drew")
    return r

⇒ https://semgrep.live/8dJ

Full Solution: https://semgrep.live/vWX
func main() {
    http.HandleFunc("/index", Handler)
    http.ListenAndServe(":80", nil)
}

https://semgrep.live/L18G

Solution: https://semgrep.live/Or8R
more tutorial goodness at
semgrep.live/learn
Tech Talk
Regex-Based Lints

Easy, but dumb

AST-Based Lints

Whole-Program Analysis

Powerful, but complex

https://instagram-engineering.com/static-analysis-at-scale-an-instagram-story-8f498ab71a0c
xkcd 1171

IF YOU’RE HAVIN’ PERL PROBLEMS I FEEL BAD FOR YOU, SON—

I GOT 99 PROBLEMS,

SO I USED REGULAR EXPRESSIONS.

NOW I HAVE 100 PROBLEMS.
Code is not a string, it’s a tree

string ≠ tree

```python
@app.route("/index")
def index():
    rep = response.set_cookie(name(), secure=False, s=func())
    return rep
```

```python
@app.route("/index")
def index():
    name(), func()
    response.set_cookie(
    return rep
```
Many tree matching tools: Gosec, Golint, Bandit, Dlint, ESLint, Flake8, Pylint, RuboCop, TSLint, and more!

Have to become an **expert in every AST syntax** for every language your team uses

Need **programming language expertise** to cover all idioms: languages have “more than one way to do it”

**Commercial SAST tools?**
- Complicated
- Slow (not CI friendly)
- Expensive

Find calls to `eval()` in only 307 LOC 👍

https://github.com/eslint/eslint/blob/master/lib/rules/no-eval.js
https://instagram-engineering.com/static-analysis-at-scale-an-instagram-story-8f498ab71a0c
In Practice
How I write rules

1. From recent post mortems: what code issues contributed to it?
2. [XYZ] is a (security, performance, other) library that everyone should use, but they don't consistently.
3. When you review code, what changes do you frequently ask for?
4. What vulnerability classes from bug bounty submissions reoccur (or appear in different places of the codebase)?
5. Are there eng / perf patterns? Consistent exception handlers?
6. What issues were caused by misconfigurations in Infrastructure-as-Code files (JSON)?
7. What are some "invariants" that should hold about your code - things that should always or never be true (e.g. every admin route checks if user is admin)?
8. What methods/APIs are deprecated and you're trying to move away from?
Order of API Calls Must be Enforced

/*
 * In this financial trading application, every transaction
 * MUST be verified before it is made
 *
 * Specifically: verify_transaction() must be called on a transaction
 * object before that object is passed to make_transaction()
 */

⇒ https://semgrep.live/6JqL
func (a *App) initializeRoutes() {
    a.Router.HandleFunc("/products",
        a.getProducts).Methods("GET")
}
Semgrep application: code inventory
Use of Weak RSA Key

// Insufficient bit size
pvk, err := rsa.GenerateKey(rand.Reader, 1024)

// Sufficiently large bit size
pvk, err := rsa.GenerateKey(rand.Reader, 2048)

⇒ https://semgrep.live/zdRI

Full Solution: https://semgrep.live/zdRI | docs
r2c provides a repository of rules for semgrep written by our team and the community, organized by languages and frameworks. This page is a web frontend for the repo with the rules; you can filter the registry here and pick rules to use in your own custom rule pack. You will then be able to use your pack by just passing a `--config` argument to Semgrep when running locally.

---

**Languages**
- javascript
- python
- c
- go
- java

**Severity**
- error
- warning
- info

**Categories**
- nodejs
- correctness
- performance
- dlint
- best-practice
- compatibility
- security
- maintainability

**Keywords or search string**

---

- `go.jwt-go.security.jwt.hardcoded-jwt-key`
- `...gorilla.security.audit.handler-assignment-from-multiple-sources...`
- `...handler-attribute-read-from-multiple-sources...`
- `...otto.security.audit.dangerous-execution...`
- `...gRPC.security.grpc-server-insecure-connection...`
NodeJsScan

v4+ powered by
Semgrep

Releasing nodejsscan v4 ⚡
Powered by semantic aware semgrep from @r2cdev with over 70+ improved rules. Try now
semgrep rules by Damian Gryski, author of *Go-Perfbook*
Awesome Use Cases

Search your code
- Vulnerabilities
- Audit security hotspots
- Extract routes
- Codify domain knowledge

Guard your code
- Secure defaults
- Banned APIs
- Best- and required-practices
- Configuration file auditing

Upgrade your code
- Migrate from deprecated APIs
- Apply automatic fixes

Watch your code
- See fixes over time
1. semgrep.live/learn
2. $(brew or pip) install semgrep
3. $ semgrep --config=r2c .
4. https://r2c.dev/survey pls :)