SCALING CODE ANALYSIS ACROSS AN ENTERPRISE

FLORIAN NOEDING

PRINCIPAL SECURITY ARCHITECT @ ADOBE

ABOUT ME

- Principal Security Architect @ Adobe

  Software Engineer → Security Researcher

  → Security Strategy

- Fun fact: I bake my own German style bread

  recipe on my blog https://florian.noeding.com

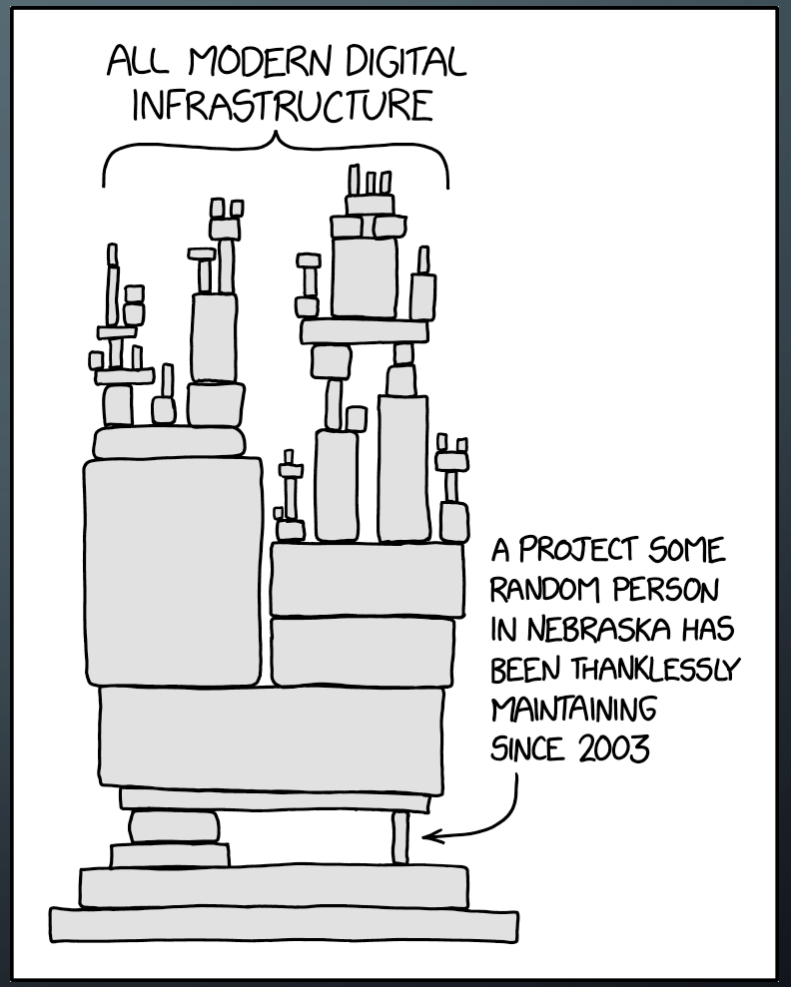
SECRETS IN SOURCE CODE (SISC)

- Detect credentials in source code or repositories

```
import requests

token = '44AE90194399'

url = 'https://example.com/api'

headers = {
    'Authorization': f'Bearer {bearer_token}'
}

response = requests.get(url, headers=headers)
```

SOFTWARE COMPOSITION ANALYSIS (SCA)

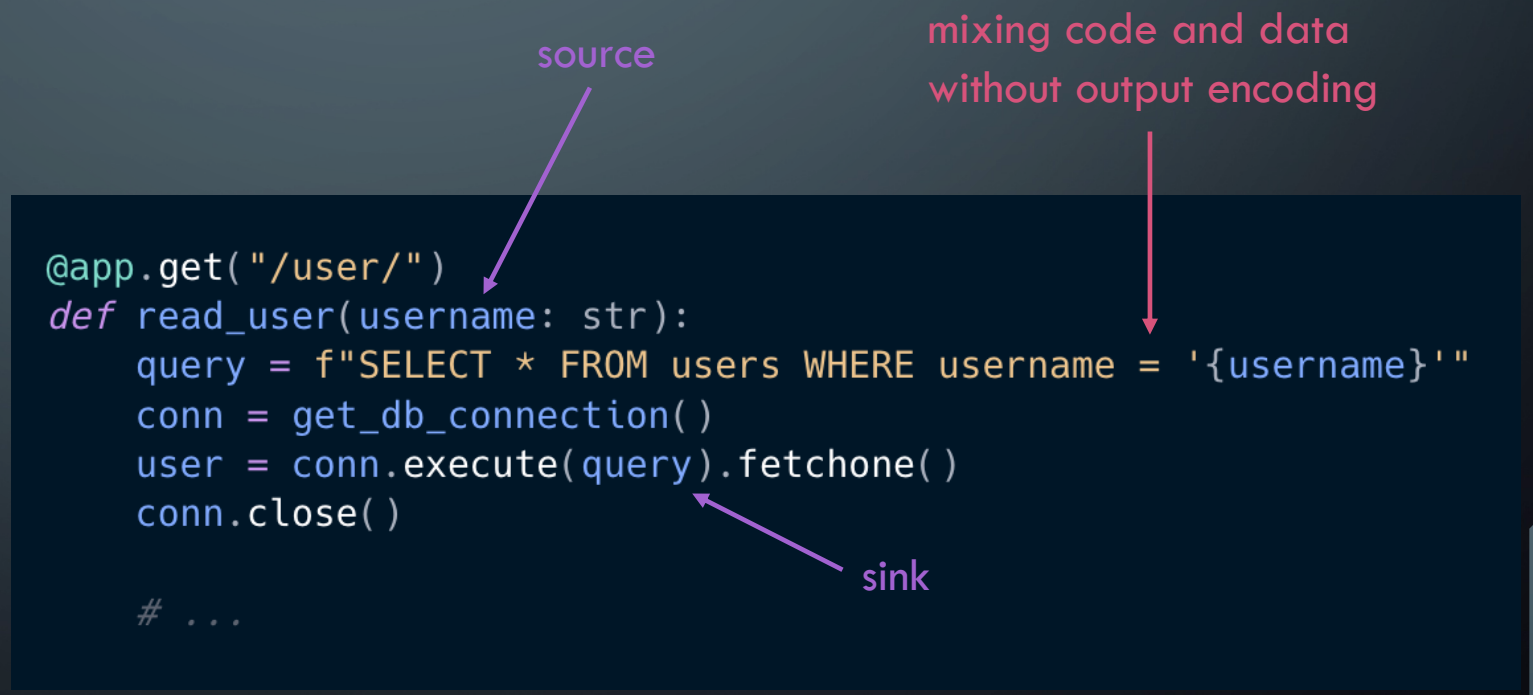
- Create inventory of 3rd party dependencies
- Enables look-up of CVEs affecting these libraries



https://xkcd.com/2347/

STATIC APPLICATION SECURITY TESTING (SAST)

- Looks for vulnerable code patterns or dataflows
- Identifies 1st party vulnerabilities



CHALLENGE

- Hundreds of Products across desktop, mobile, web, …
- Diverse tech stacks
  - 12 programming languages make up 80% of our code
  - Many more frameworks
- Multiple SCMs
- 100k+ repositories as primary scope
- On average 30k scan events per day

DESIGN, ROLLOUT, IMPACT

DESIGN PRINCIPLES

Great Developer Experience as the key goal to ensure acceptance:

- Integrate deeply into developer workflow
- Provide timely, relevant and actionable feedback (shift-left!)
- Carefully balance noise and risk reduction
- Single pane of glass into source code related findings

Goal: pragmatic risk reduction instead of zero known vulnerabilities.

PRIMARY PROCESS

Security as Code

Adversary Intelligence

Scan requested

Feedback
<= 5 min + build

write code

test locally

commit & push

create pull request

gather feedback

merge

FEEDBACK LOOPS

Inline feedback on pull requests	Findings in changed files only (important!), very few exceptions
Metrics driven security campaigns	only a few specific risks
Custom dashboard	All findings on any branch Challenge: not yet widely adopted by engineers
Ticketing for enforcement	Critical risks only Challenge: Attribution to project often non-trivial

SCAN TOOL SELECTION

Easy to roll-out

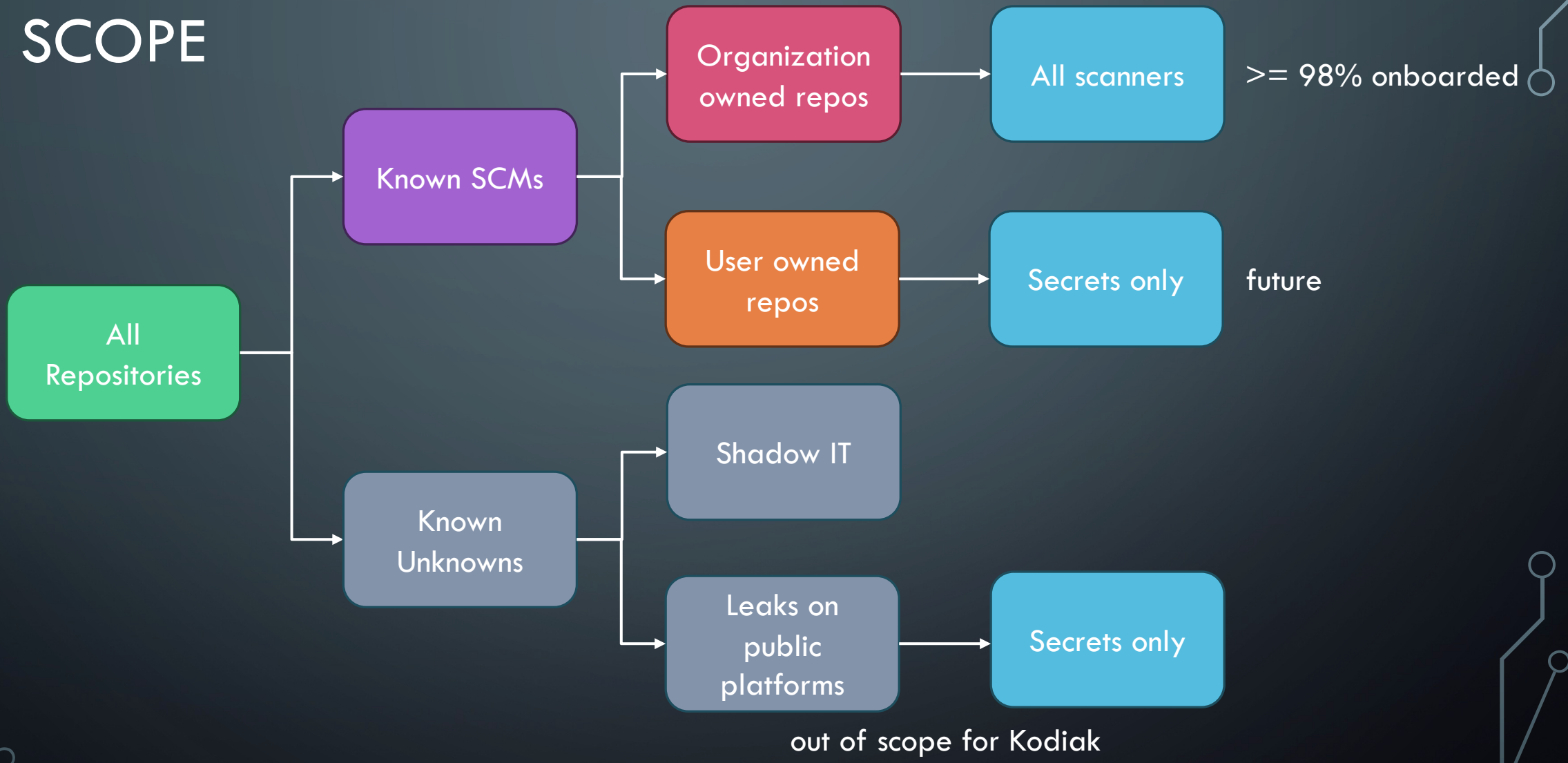
Finds important vulnerabilities

Developer friendly output

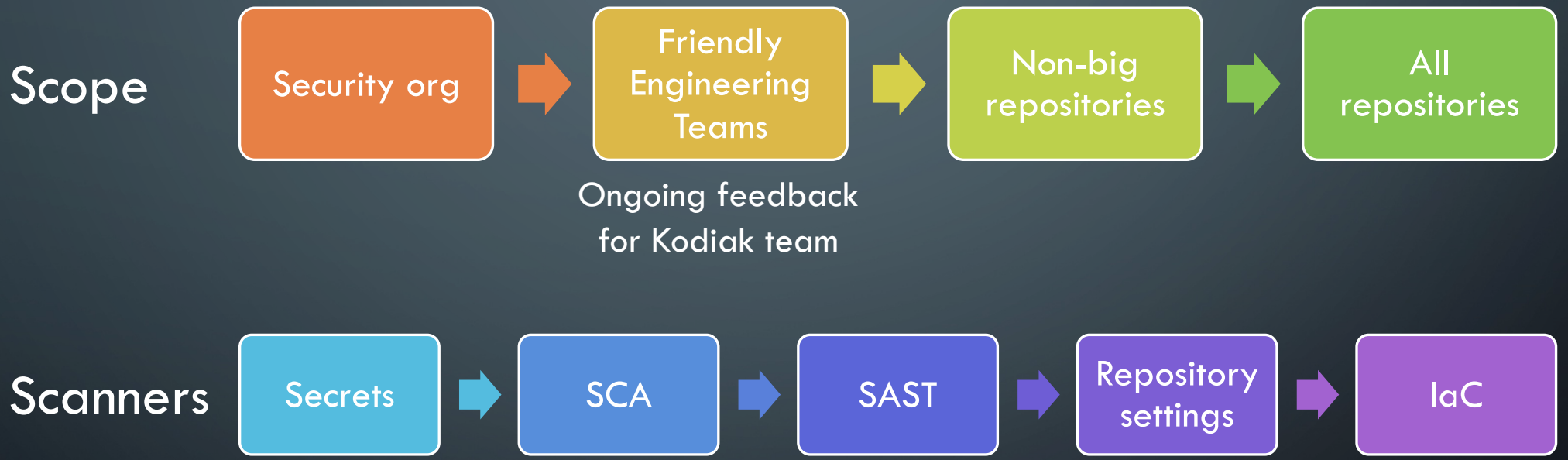
Fast enough

Customizable

SCOPE


All Repositories
Known SCMs
Known Unknowns
Organization owned repos
User owned repos
Shadow IT
Leaks on public platforms
All scanners
Secrets only
Secrets only
>= 98% onboarded
future
out of scope for Kodiak

ROLLOUT


Scope
Security org
Friendly Engineering Teams
Non-big repositories
All repositories
Ongoing feedback for Kodiak team
Scanners
Secrets
SCA
SAST
Repository settings
IaC

OUTCOMES 2023

300,000 findings fixed

nudging only – zero enforcement

RISK PRIORITIZATION

ADVERSARY MODEL

HIGHLY SIMPLIFIED

Security Researchers	eCriminals	Nation States
• Various motivations • generally friendly • no 0-day leaks	• Often financially motivated • Repeatable, scalable exploitation	• Operations with targeted outcomes • Hard to predict
→ Use to identify gaps in program	→ Fix easy to exploit or widely deployed vulnerabilities first	→ Defense in depth

SISC – RISK REDUCTION STRATEGY

MOST BREACHES INVOLVE LEAKED OR STOLEN CREDENTIALS

SCA – RISK REDUCTION STRATEGY

MANY BREACHES START WITH AN OUT-OF-DATE SYSTEM

Future:

- Filter out unreachable CVEs
- Use contextual data

SBOM Transparency:

→ Fix based on CVSS

SAST – RISK REDUCTION STRATEGY

OPEN PROBLEM – RELYING ON SAST VENDOR'S SCORES

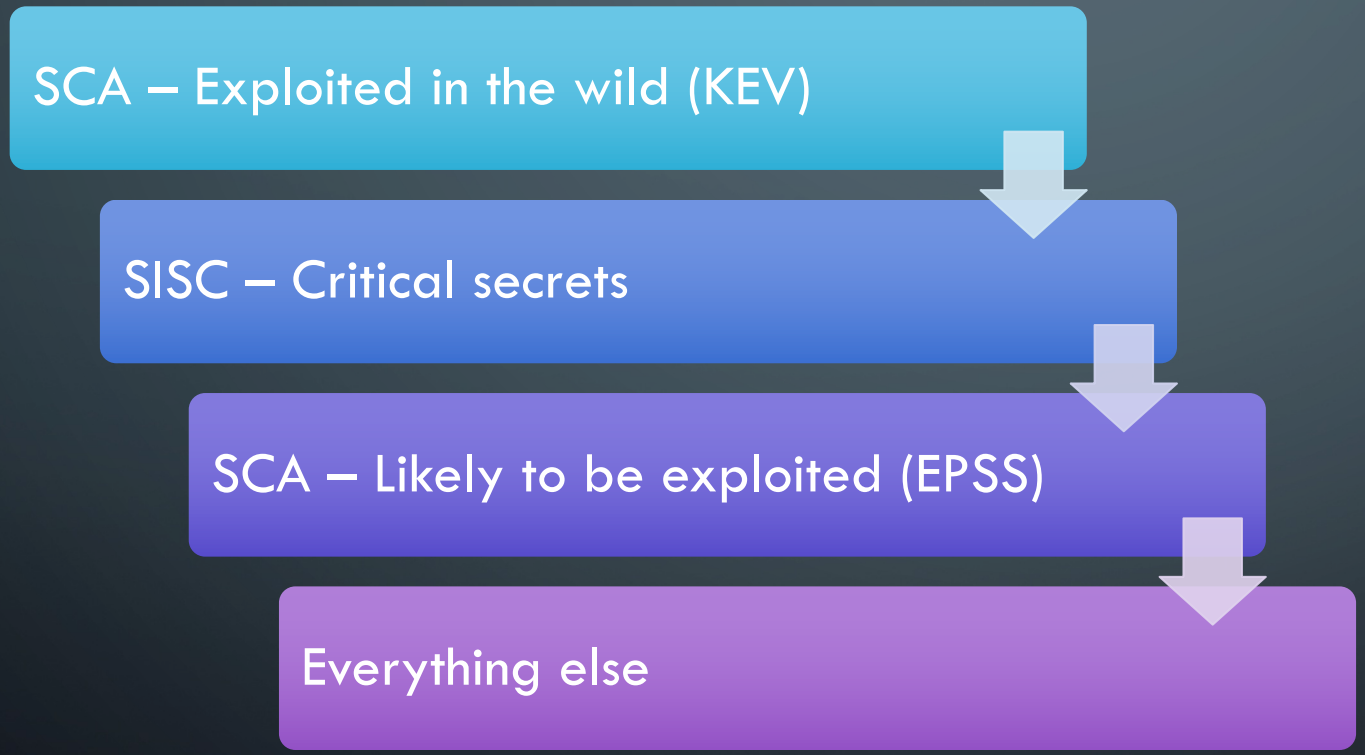
Future:

- CWE ⇔ TTP mapping?

Let me know if you've solved this!

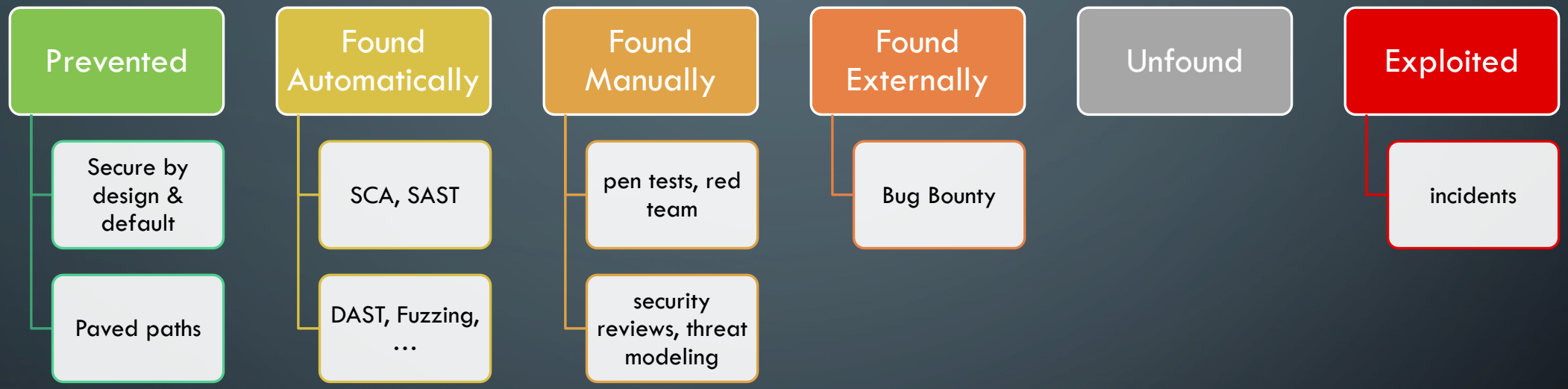
UNIFIED RISK PRIORITIZATION

SIMPLIFIED MODEL



META FEEDBACK LOOP

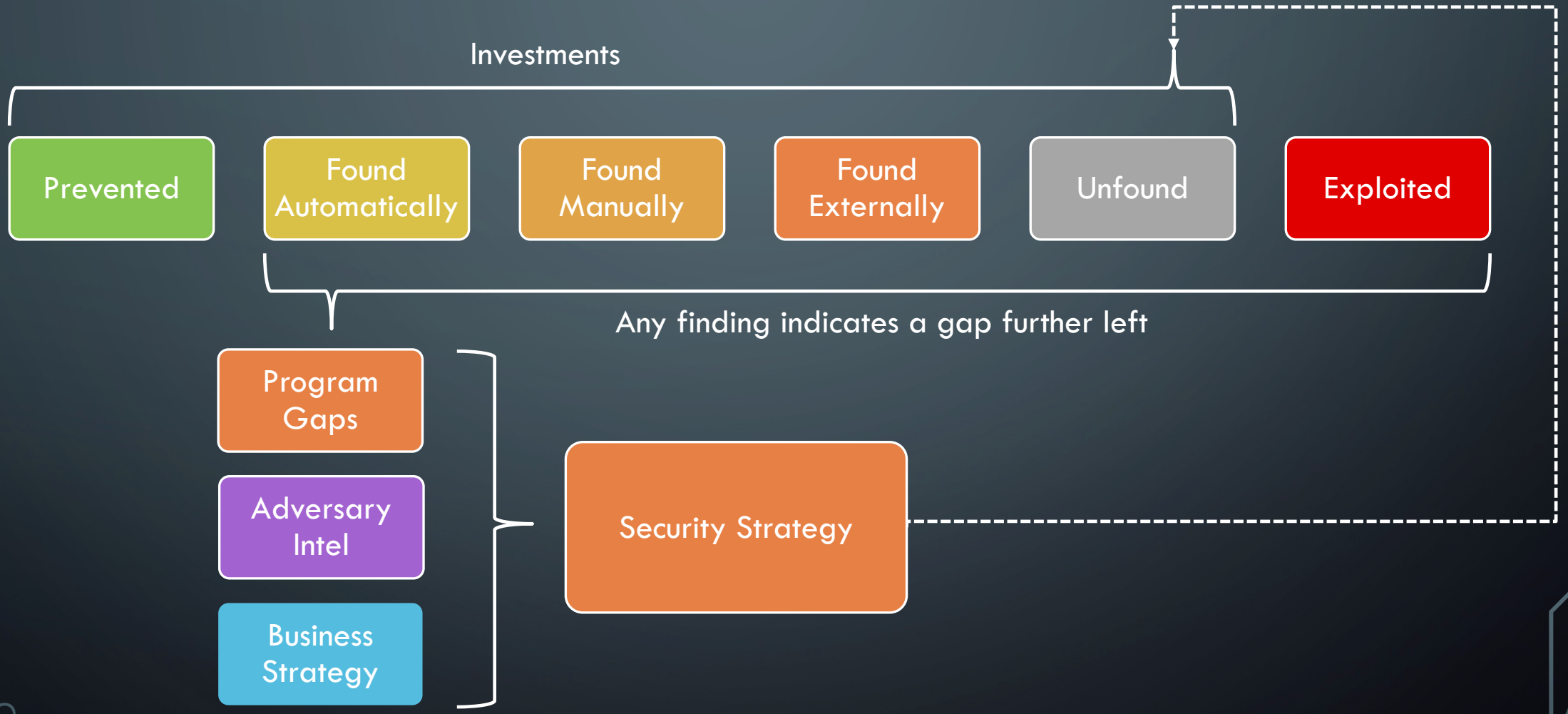
SHIFTING LEFT – 6 BUCKETS OF RISK



Fixing root causes > Fixing symptoms

Hazard elimination > Hazard remediation

META FEEDBACK LOOP



QUESTIONS?

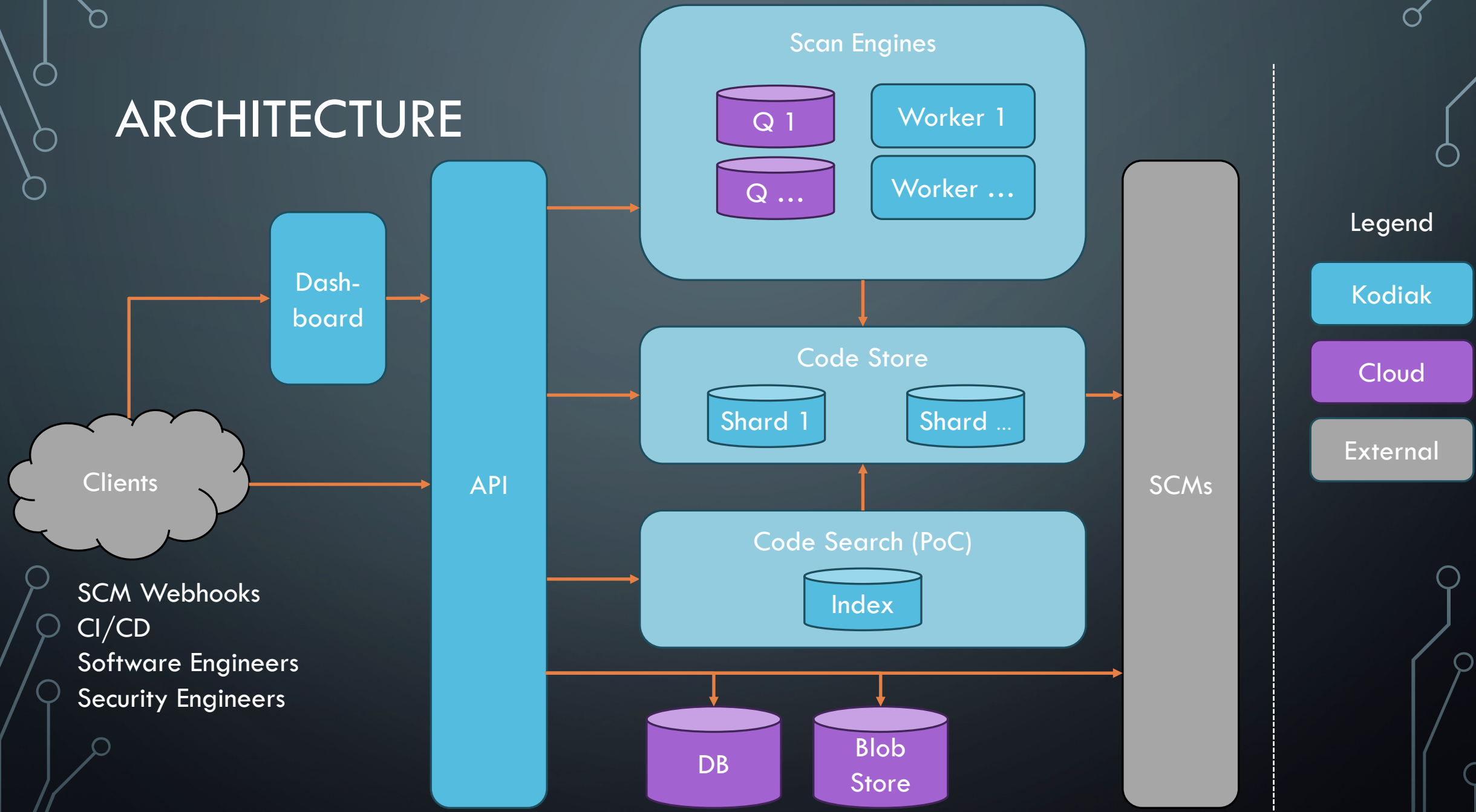
Key Takeaways:

- Focus on great DevEx – talk to them!
- Feedback loop design is crucial
- Fix things that matter
  - SCA: Consider using EPSS
  - SAST: target root causes, not symptoms

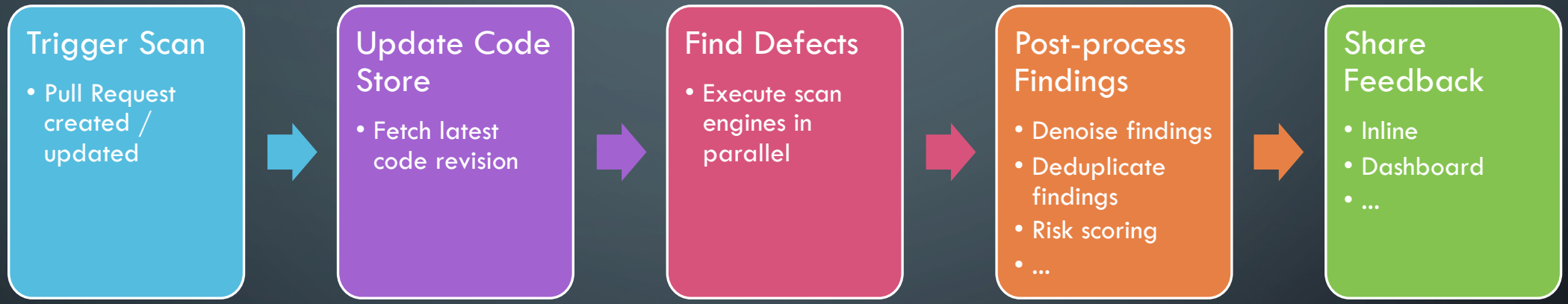
https://florian.noeding.com

BACKUP SLIDES

ARCHITECTURE



KODIAK INTERNALS



LINKS

- EPSS: https://www.first.org/epss/
- KEV: https://www.cisa.gov/known-exploited-vulnerabilities-catalog
- 6 buckets of risk: http://collingreene.com/6_buckets_of_prodsec.html
- My blog: https://florian.noeding.com
- More about Kodiak: https://blog.developer.adobe.com/project-kodiak-shifting-application-security-left-at-enterprise-scale-55f5453d1966