FQTTERCL-JLVA02-OBC XTNQUE[]F

Daniel Cuthbert | Mark Carney | Benjamin Rodes | Niroshan Rajadurai

December 2023

Hello

Daniel “Marty” Cuthbert

Global Head of Security Research

Mark “Mother” Carney

Senior Researcher, CTO @ Quantum Village

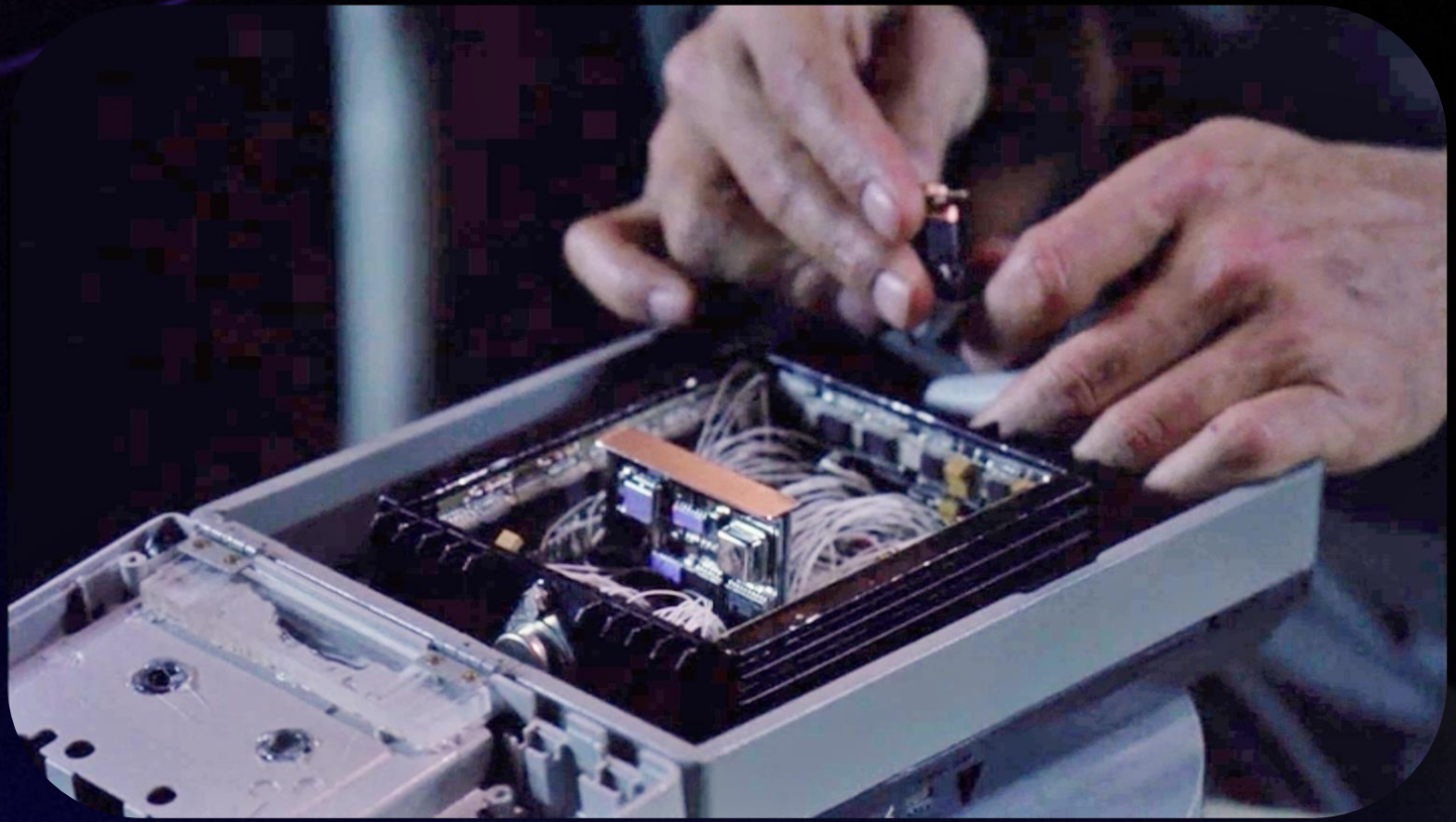
Benjamin “Whistler” Rodes

Principal Security Engineer, Microsoft

Niroshan “Donald” Rajadurai

Sr. Director, GitHub Advanced Security & AI

Crypto != Cryptography



So why does this matter?

RFC 6320

```
grep -r -E '\b([Hh][Mm][Aa][Cc]-)?[Mm][Dd]5\b' /supersecurecode/*
```

Why do we have this?

From: LogJam-CVSS-of-4.0-honest-please-fix-me-draft-draft-FINAL.docx

To:
Quantum computing


Q-Day is
coming...

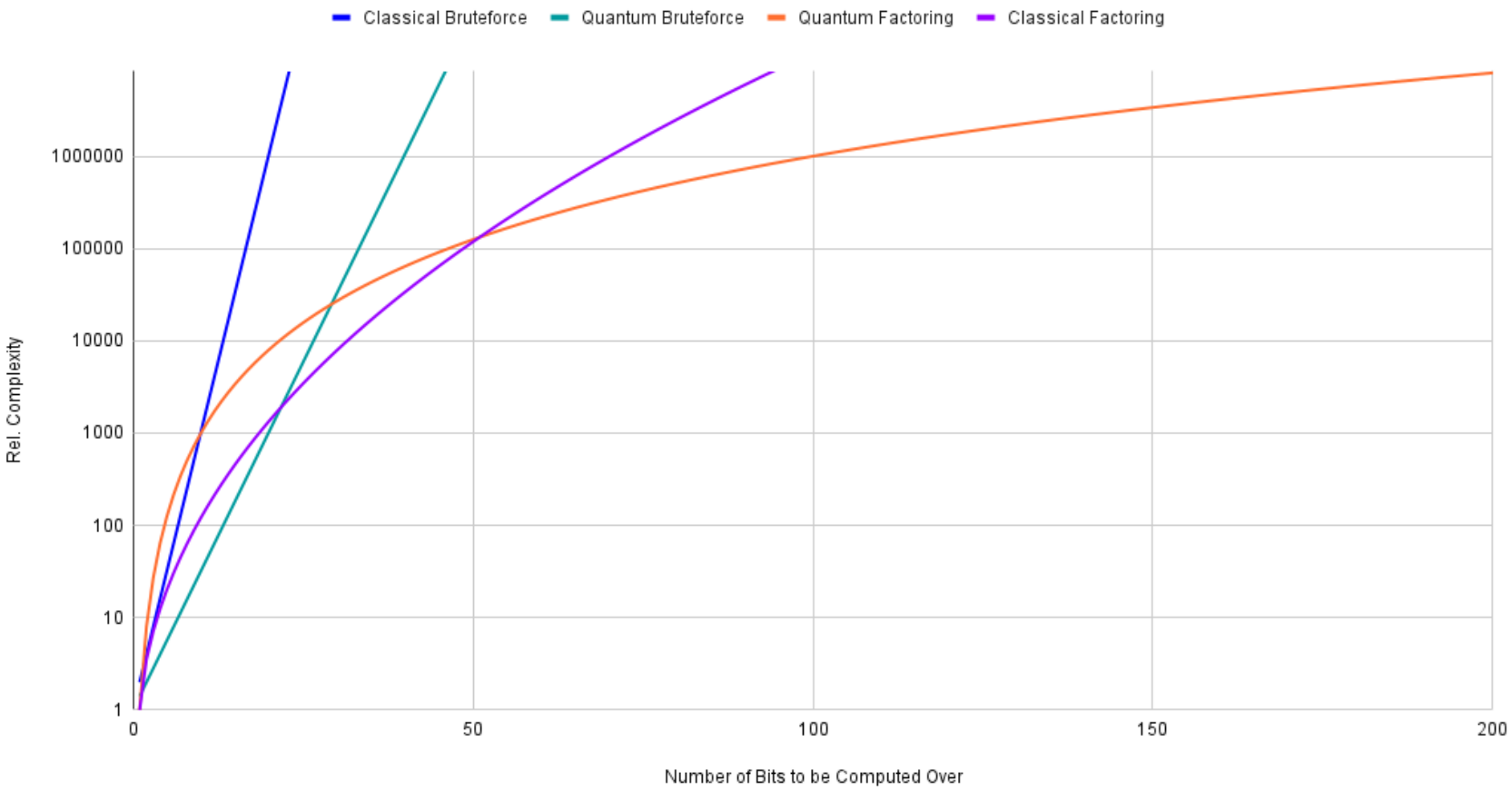
APRIL 14, 2030

Source: cloudsecurityalliance.org

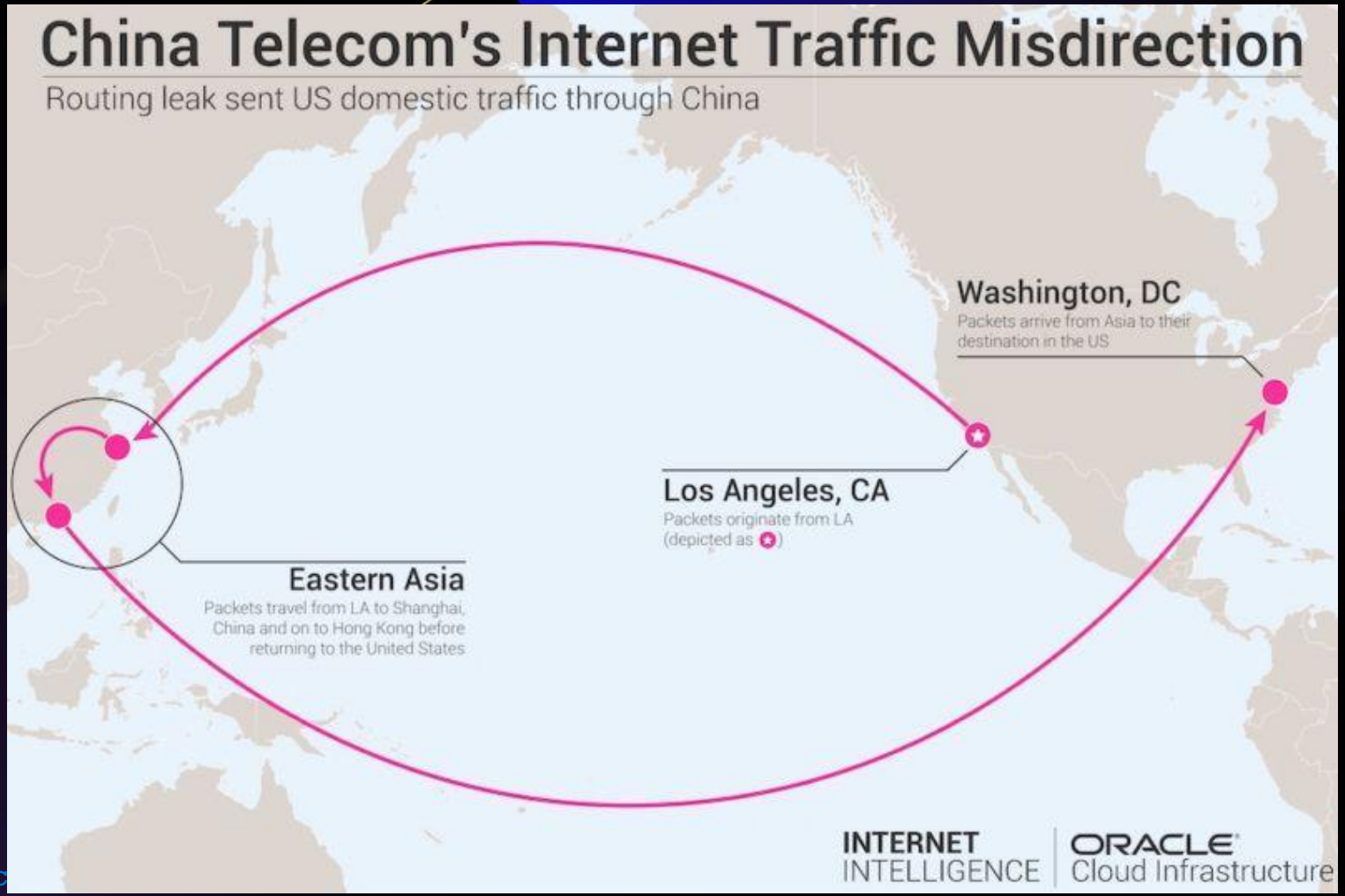
What does that mean for cryptography?

Quantum vs. Classical Hardness

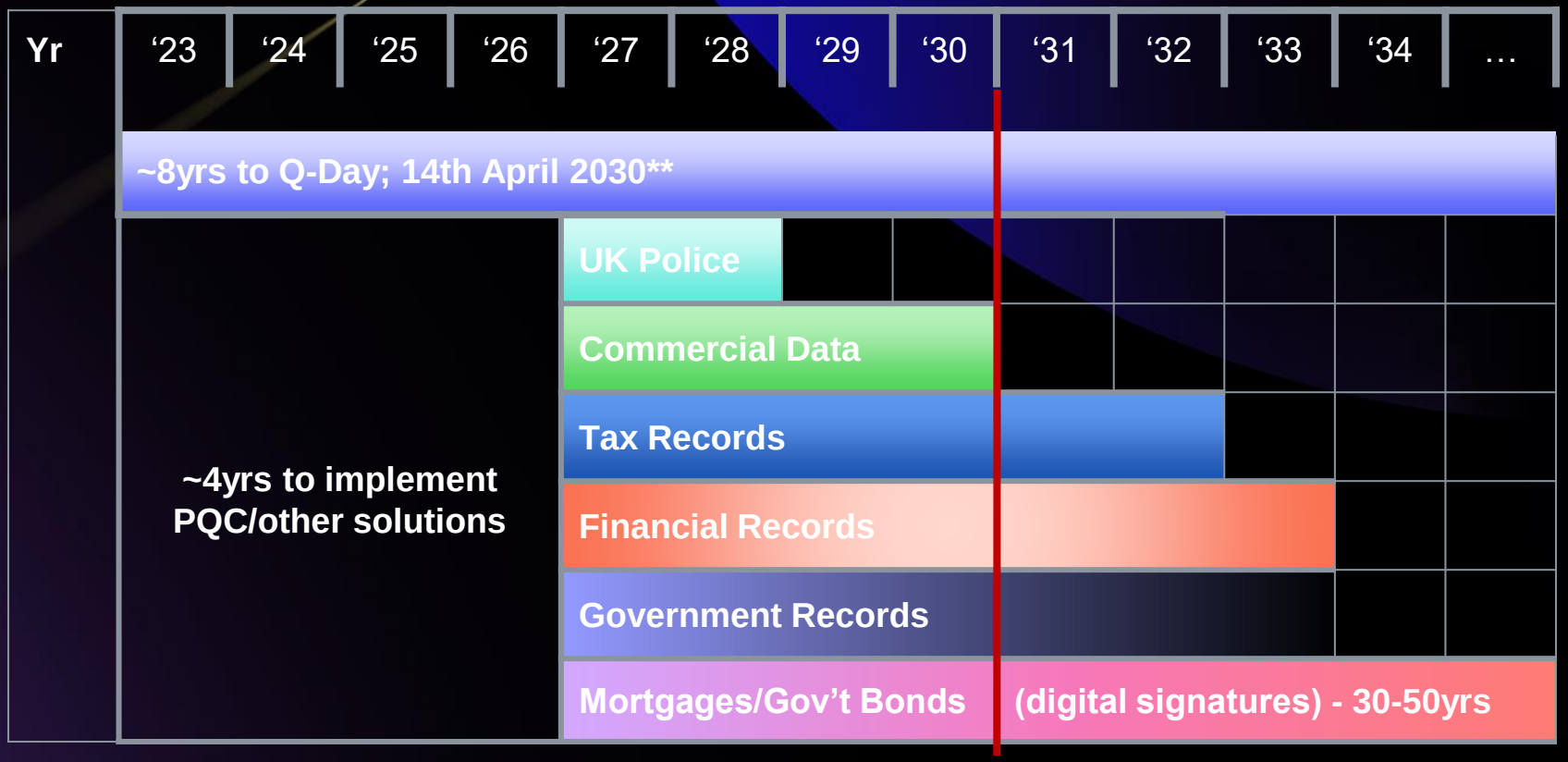
Curved and less vertical lines indicate lower complexity/hardness



Infor


China Telecom's Internet Traffic Misdirection
Routing leak sent US domestic traffic through China
Washington, DC
Packets arrive from Asia to their destination in the US
Los Angeles, CA
Packets originate from LA (depicted as)
Eastern Asia
Packets travel from LA to Shanghai, China and on to Hong Kong before returning to the United States
INTERNET INTELLIGENCE
ORACLE Cloud Infrastructure

“It’s all about the information”


Yr
‘23
‘24
‘25
‘26
‘27
‘28
‘29
‘30
‘31
‘32
‘33
‘34
…
~8yrs to Q-Day; 14th April 2030**
~4yrs to implement PQC/other solutions
UK Police
Commercial Data
Tax Records
Financial Records
Government Records
Mortgages/Gov’t Bonds
(digital signatures) - 30-50yrs

So we have to prepare for tomorrow today

Creating an environment for cryptographic agility

LOCATE

COMPARE

DETERMINE

REPLACE

MONITOR

Where do we start?

Focus on cryptographic agility, the rest will follow

LOCATE

COMPARE

DETERMINE

REPLACE

MONITOR

“describes steps for agencies to undertake as they begin their transition to PQC [post quantum cryptography] by conducting a prioritized inventory of cryptographic systems.”

White House Memorandum M-23-02

CBOM

(Cryptographic Bill Of Materials)

A record containing the details of various cryptographic software components used in a software system

Why is CBOM Generation Complex?

01 **API Variability**

What's the "space" of possibilities?

02 **Data Flow Complexity**

How do we analyze this space?

03 **Cryptography Abstractions & Modeling APIs**

How do we codify (*model*) the analysis for each API use?

Data Flow Example: Finding Key Gen Config

```c
void foobar(int size){
    EVP_PKEY_CTX *ctx;
    EVP_PKEY *pkey = NULL;

    ctx = EVP_PKEY_CTX_new_id(EVP_PKEY_RSA, NULL);
    if (!ctx) {
        /* Handle error */
    }

    if (EVP_PKEY_CTX_set_rsa_keygen_bits(ctx, size) <= 0) {
        /* Handle error */
    }

    if (EVP_PKEY_keygen(ctx, &pkey) <= 0) {
        /* Handle error */
    }

    /* Do something with pkey */
}
```

May be from multiple sources or “unknown”

Trace data to this to variable to find key size

Data Flow Example: Finding Default Configuration

```c
void foobar(){
    EVP_PKEY_CTX *ctx;
    EVP_PKEY *pkey = NULL;

    ctx = EVP_PKEY_CTX_new_id(EVP_PKEY_RSA, NULL);
    if (!ctx) {
        /* Handle error */
    }

    // if (EVP_PKEY_CTX_set_rsa_keygen_bits(ctx, 2048) <= 0) {
    //     /* Handle error */
    // }

    if (EVP_PKEY_keygen(ctx, &pkey) <= 0) {
        /* Handle error */
    }

    /* Do something with pkey */
}
```

The set_*rsa_keygen_bits* operation is not required!

Does an rsa_keygen_bits result flow here?
If not, what algorithm does CTX represent?
This algorithm would have a default/unknown key size.

CodeQL

GitHub’s static analysis engine powered by curated custom queries to hunt for vulnerabilities in your code

Supports a wide range of languages

C/C++, C#, Go, Java, Kotlin, JavaScript, Python, Ruby, Swift, TypeScript

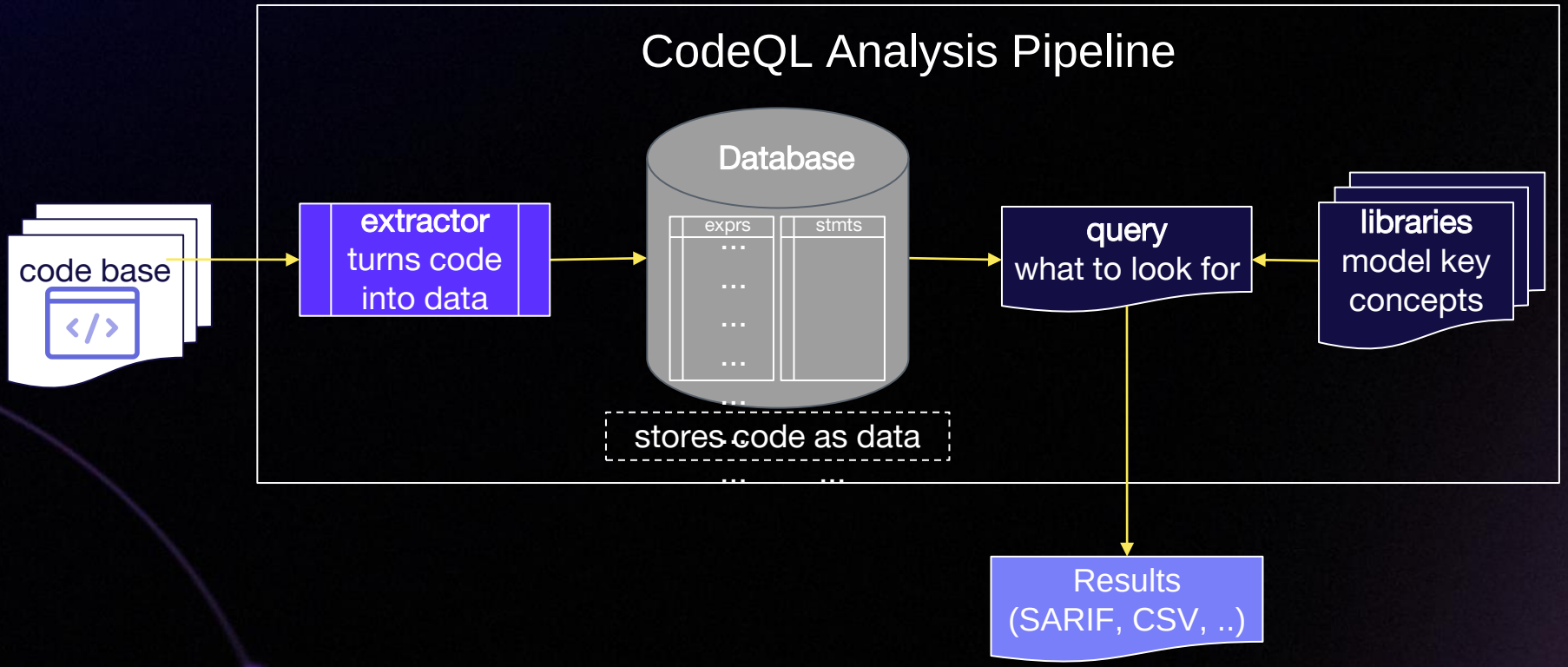
Robust static analyses

Including interprocedural data flow

Works at scale

Open source

How CodeQL works



Leveraging CodeQL for CBOM Generation

Simple, informative queries leveraging cryptography abstractions

```
from BlockModeAlgorithm alg
select alg, "Use of algorithm " + alg.getBlockModeName()
```

```
from EllipticCurveAlgorithm alg
select alg, "Use of algorithm " + alg.getCurveName()
```

Abstract Class

```
from AsymmetricAlgorithm alg
select alg, "Use of algorithm " + alg.getName()
```

```
from BlockModeAlgorithm alg
select alg.getIVorNonce(), "Block mode IV/Nonce source"
```

```
from HashAlgorithm alg
select alg, "Use of algorithm " + alg.getName()
```

```
from AsymmetricKeyGeneration op, CryptographicAlgorithm alg, Expr configSrc
where
  alg = op.getAlgorithm() and
  configSrc = op.getKeyConfigurationSource(alg)
select op, "Key generator for algorithm $@ with key configuration $@", alg, alg.getName(),
  configSrc, configSrc.toString()
```

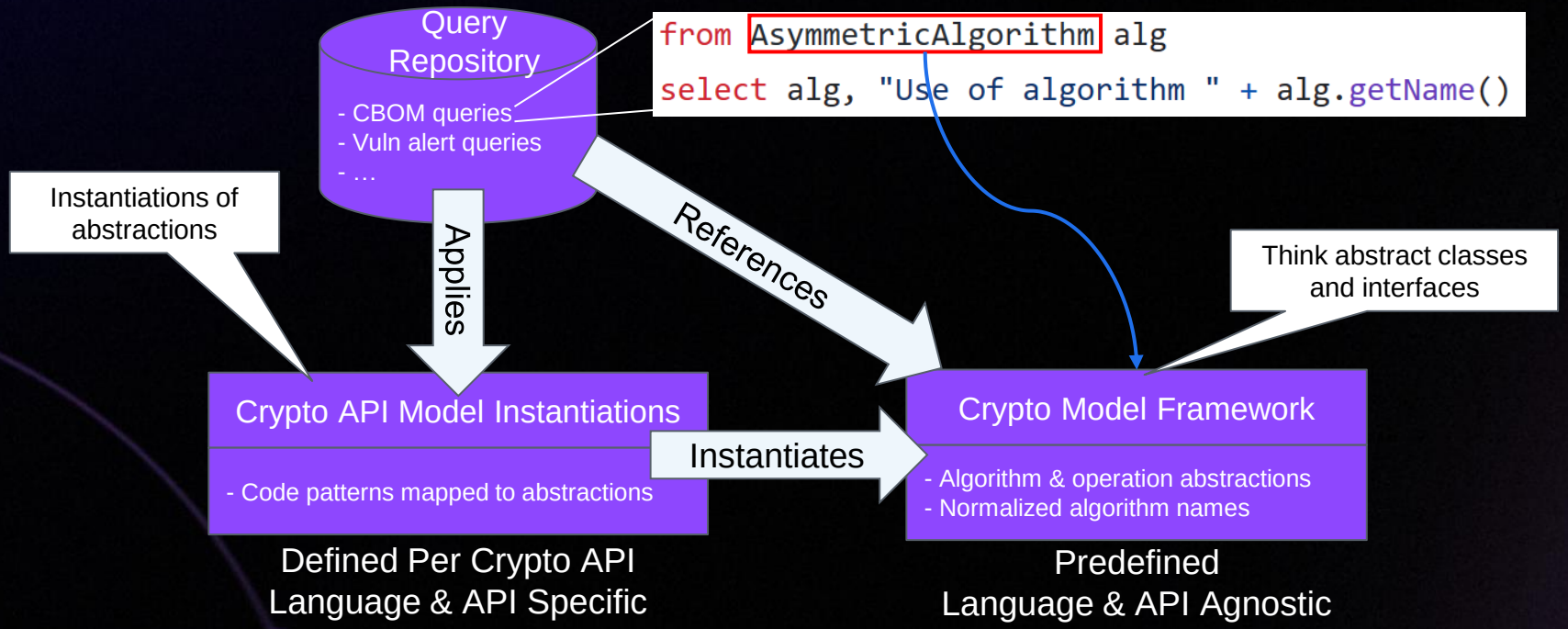
Unlocking additional information becomes trivial

Same abstractions used for CBOM

```
from AsymmetricKeyGeneration op, AsymmetricAlgorithm alg, Expr configSrc, int size
where
  alg = op.getAlgorithm() and
  not alg instanceof EllipticCurveAlgorithm and
  configSrc = op.getKeyConfigurationSource(alg) and
  size = configSrc.getValue().toInt() and
  size < 2048
select op,
  "Use of weak asymmetric key size (in bits) " + size + " configured at $@ for algorithm $@",
  configSrc, configSrc.toString(), alg, alg.getName().toString()
```

Added threshold of an *'acceptable'* size threshold (alerts if the size is <2048)

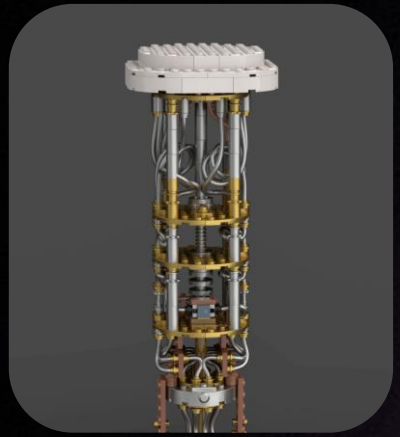
Cryptography Modeling Architecture



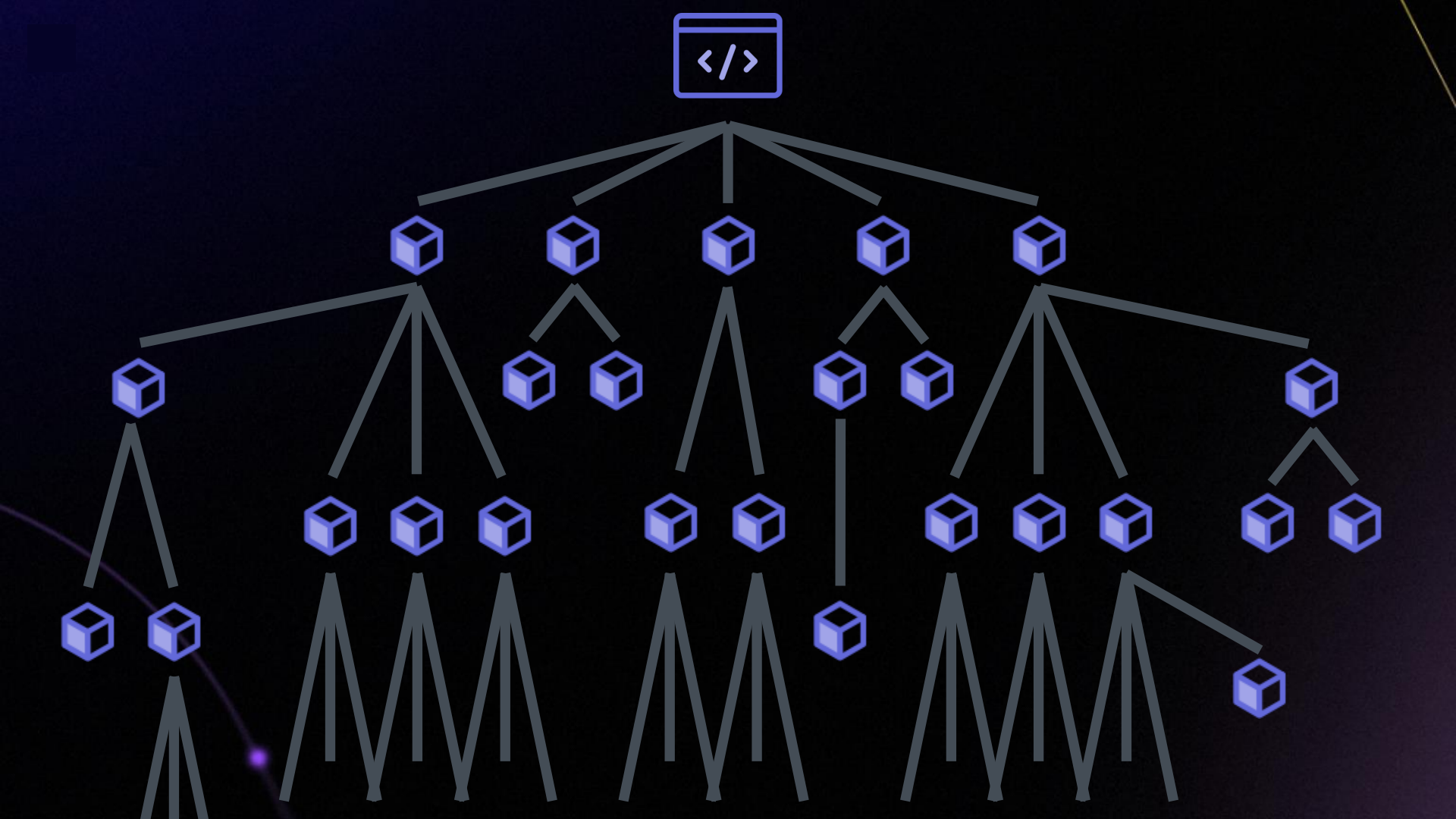
Connecting the pieces

Start with our why

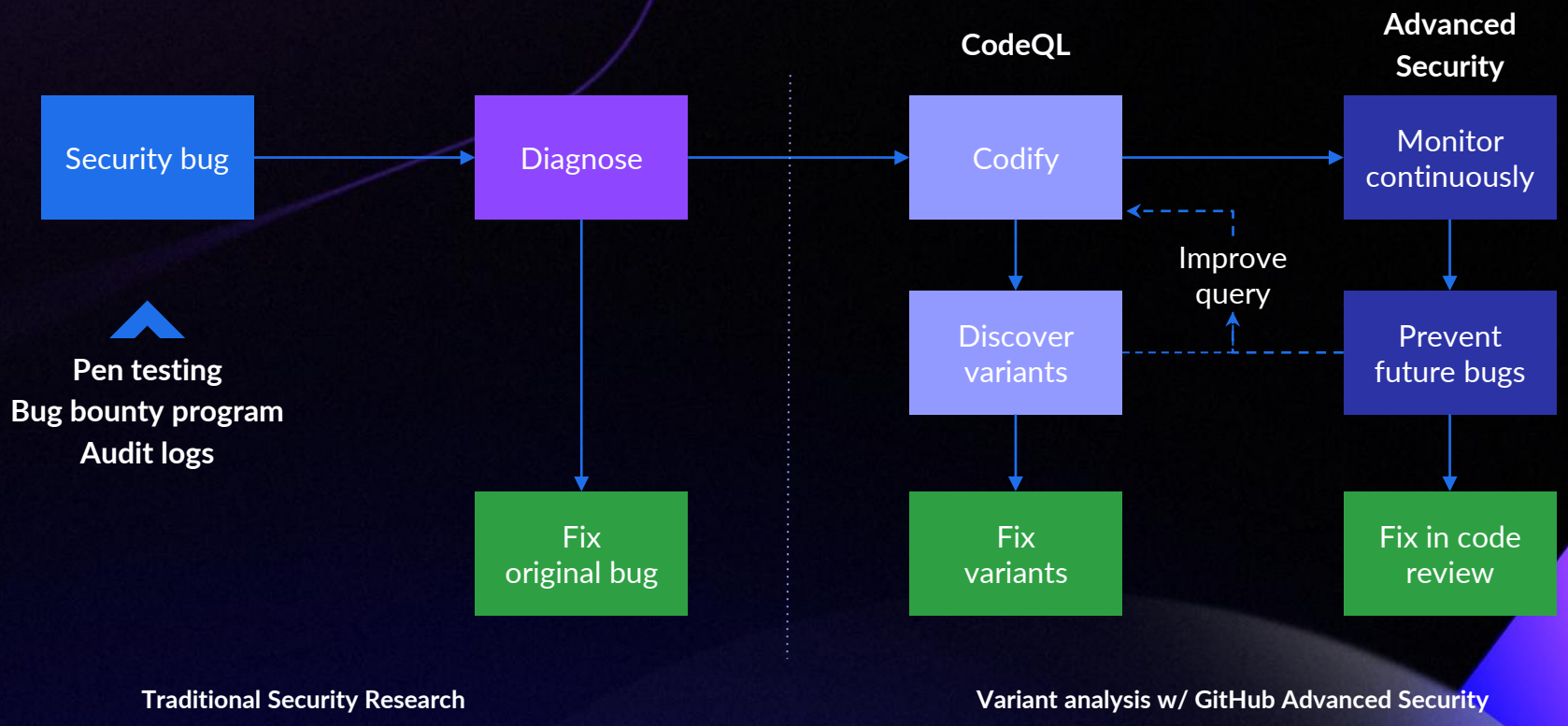
Source the information



Transform it to provide meaningful context

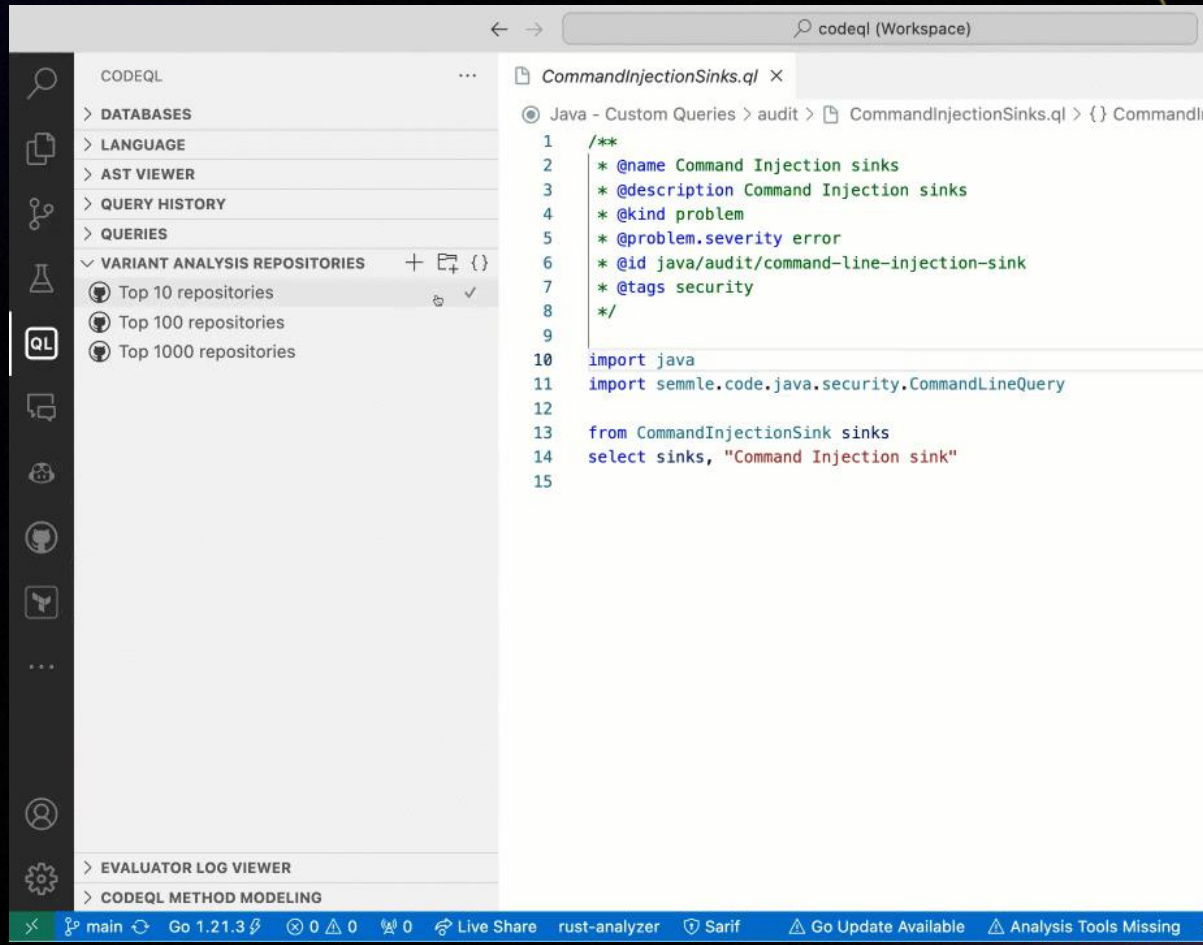


Understanding Variant Analysis



Threat hunt at scale

GitHub's CodeQL Multi Repository Variant Analysis (MRVA)



CBOM Reporter

https://github.com/santandersecurityresearch/cryptobom-forge

Generates a CBOM in CycloneDX standard to identify and enumerate cryptographic assets and vulnerabilities in a repository from the CodeQL PQC query output.

sarif.py | algorithm.py | cryptocheck.py

```python
    algorithm_component = Component(
        bom_ref=f'cryptography:algorithm:{uuid.uuid4()}',
        name=name,
        type=ComponentType.CRYPTO_ASSET,
        crypto_properties=crypto_properties
    )

    if not (existing_component := _is_existing_component_overlap(cbom, algorithm_component)):
```

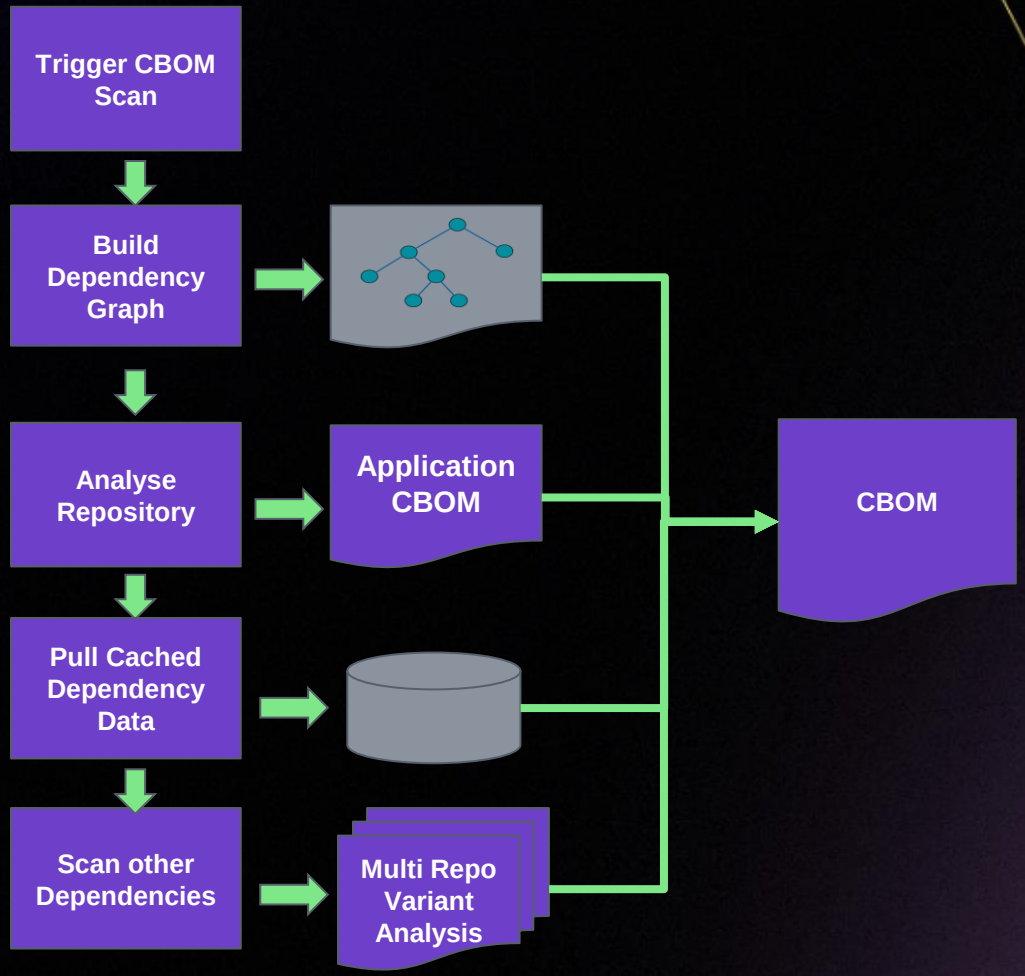
_infer_primitive()

Terminal | Local


```
      },
      {
        "ruleId": "ECDSA-detect",
        "ruleIndex": 0,
        "message": {
          "text": "ECDSA was found in use, which is not Post-Quantum Safe and should be flagged for migration to PQC a
lgorithms when they become available."
        },
        "level": "note",
        "locations": [
          {
            "physicalLocation": {
```

```
    "bom-ref": "cryptography:algorithm:89fb775a-fe21-4327-b024-d6572da3decb",
    "cryptoProperties": {
      "algorithmProperties": {
        "cryptoFunctions": [
          "digest",
          "sign"
        ],
        "padding": "unknown",
        "primitive": "hash",
        "variant": "MD5"
      },
      "assetType": "algorithm",
```

Applying multi repository variant analysis to CBOMs



Create Crypto Bill of Materials #1

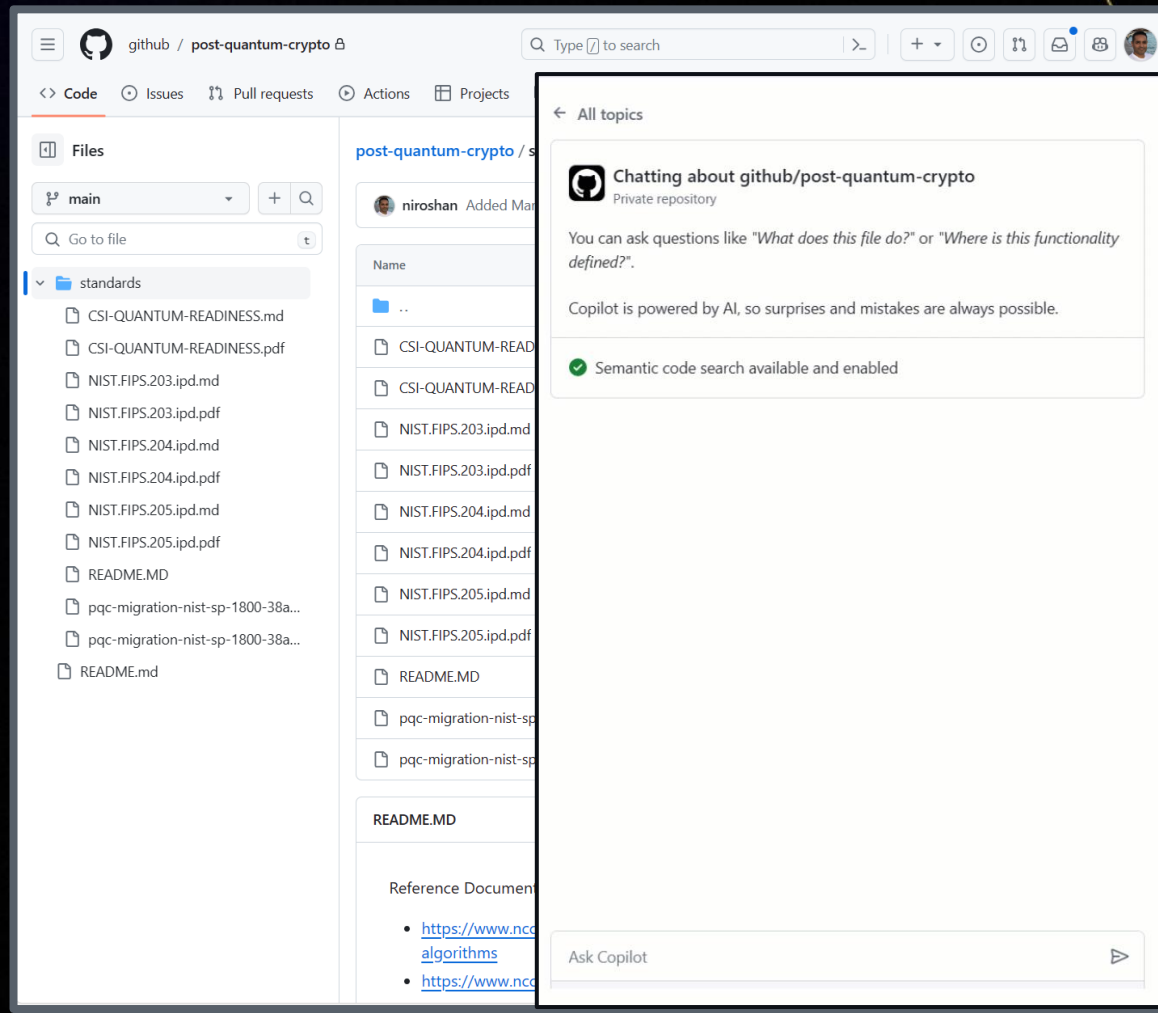
Success

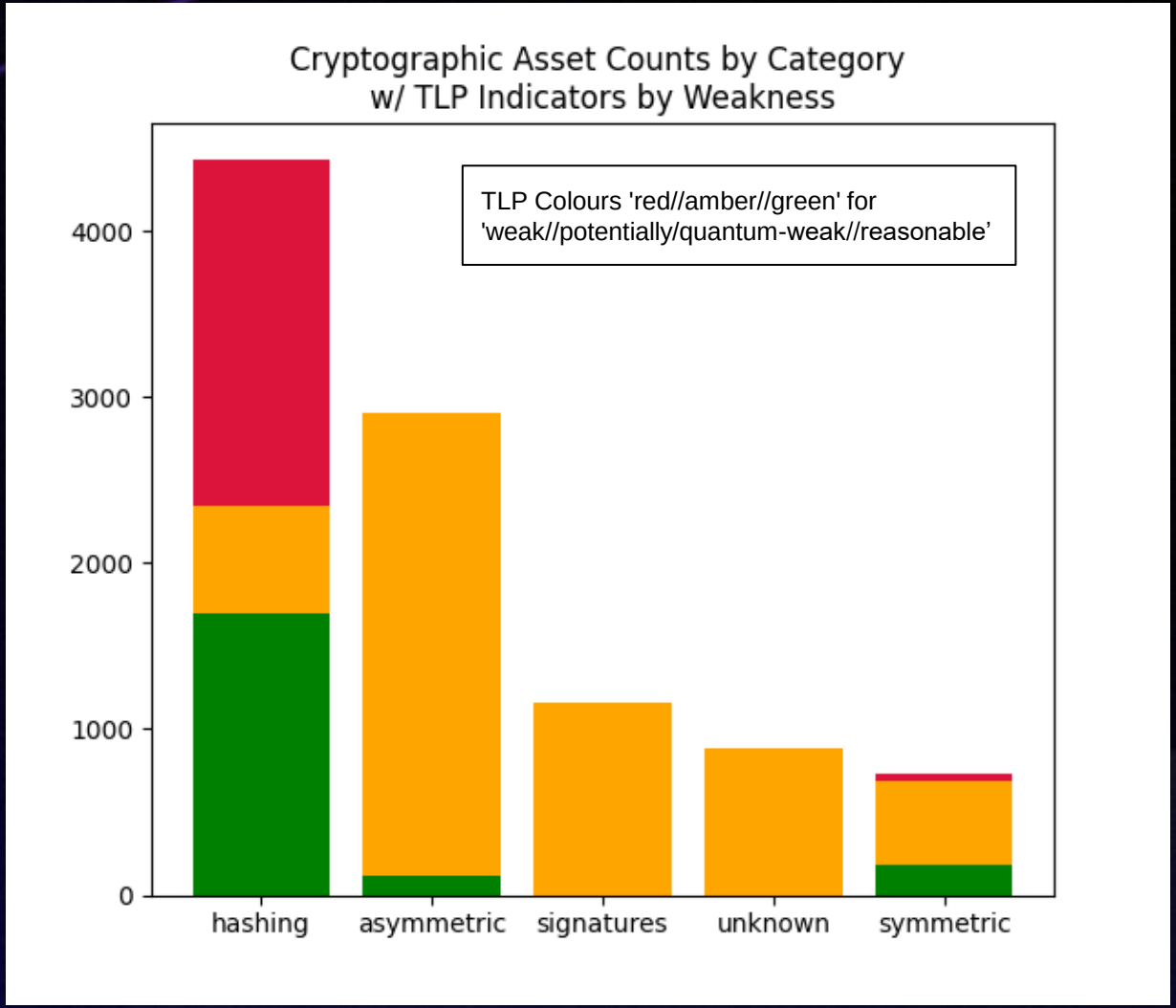
Artifacts

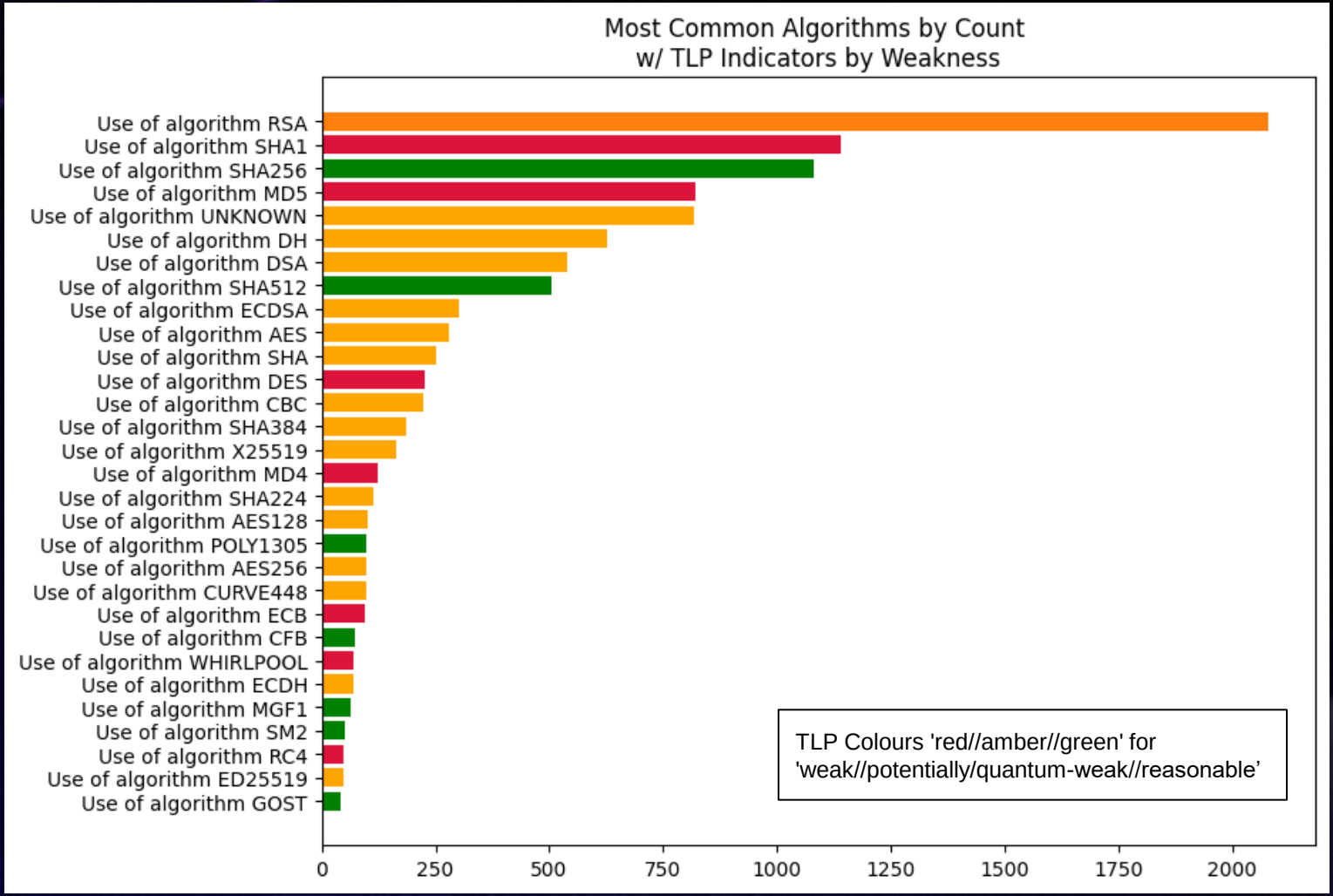
CSCM results summary

Information to drive action

GitHub's Copilot leverages Retrieval Augmented Generation (RAG) techniques to allow tailored coaching within the business on specific internal Cyber Strategies.




Cryptographic Asset Counts by Category
w/ TLP Indicators by Weakness
TLP Colours 'red//amber//green' for
'weak//potentially/quantum-weak//reasonable'
4000
3000
2000
1000
0
hashing
asymmetric
signatures
unknown
symmetric


Most Common Algorithms by Count
w/ TLP Indicators by Weakness
Use of algorithm RSA
Use of algorithm SHA1
Use of algorithm SHA256
Use of algorithm MD5
Use of algorithm UNKNOWN
Use of algorithm DH
Use of algorithm DSA
Use of algorithm SHA512
Use of algorithm ECDSA
Use of algorithm AES
Use of algorithm SHA
Use of algorithm DES
Use of algorithm CBC
Use of algorithm SHA384
Use of algorithm X25519
Use of algorithm MD4
Use of algorithm SHA224
Use of algorithm AES128
Use of algorithm POLY1305
Use of algorithm AES256
Use of algorithm CURVE448
Use of algorithm ECB
Use of algorithm CFB
Use of algorithm WHIRLPOOL
Use of algorithm ECDH
Use of algorithm MGF1
Use of algorithm SM2
Use of algorithm RC4
Use of algorithm ED25519
Use of algorithm GOST
0
250
500
750
1000
1250
1500
1750
2000
TLP Colours 'red//amber//green' for
'weak//potentially/quantum-weak//reasonable'

CryptographyStandard Internal

Watch 0 | Fork 4 | Star 0

main | 3 branches | 2 tags | Go to file | Add file | Code

Merge pull request #119 from santander-group-... eac0bd8 5 days ago 252 commits

.github/workflows	trying ghcr.io pandoc image in gh action	last month
Cryptography	Merge pull request #108 from santander-group-cyber-cto/n473021-...	last week
Implementations	Merge pull request #119 from santander-group-cyber-cto/sshd-split	5 days ago
KeyManagement	fixing issues 93, 92, 91, 90	2 months ago
resources	Initial git push from CSR repo into main group EM	9 months ago
Annex.md	Create Annex.md	5 months ago
CryptographyStandard.docx	Initial git push from CSR repo into main group EM	9 months ago
Governance.md	added exception management	2 months ago
Intro.md	removed exception management	2 months ago
README.md	re-added trivy scan flare	2 months ago
changelog.md	Issue # 10	4 months ago
gen-changelog.sh	Initial git push from CSR repo into main group EM	9 months ago
index.txt	move changelog to end of doc	2 months ago

README.md

Docker Code Scanning passing

Santander Global Cryptography Standard

Please see the Intro page for details about the standard.

About

Santander Group Cryptography Standard - This document contains mandatory security requirements for the effective use of cryptography for security within Santander Group.

Readme
Activity
0 stars
0 watching
4 forks

Releases 2

v20230703 Latest
on Jul 5

+ 1 release

Packages

No packages published
Publish your first package

Contributors 7

Languages

Shell 100.0%

Securing our digital landscape takes all of us

Recap:

How we prepare for the Post Quantum Crypto world

Understand the risks

Locate and assess

CBOM

Scale your efforts

Instill Crypto Agility

Try it out
&
Help the community

Microsoft

Santander


Rasmus Larsen


Alvaro Muñoz


Chris Campbell


Walker Chabbott


Paul Hodgkinson


Rutger Schenk

Thank You


Bas van Schaik


Raul Garcia


James Fletcher


Pierre Tempel


Josh Brown White


Christina Delahanty


Emile El-Qawas

Try it out
&
Help the community