SBOM/TAIBOM what are they are about
Oxford 19/11/24

Dr Nicholas Allott
nick@nqminds.com

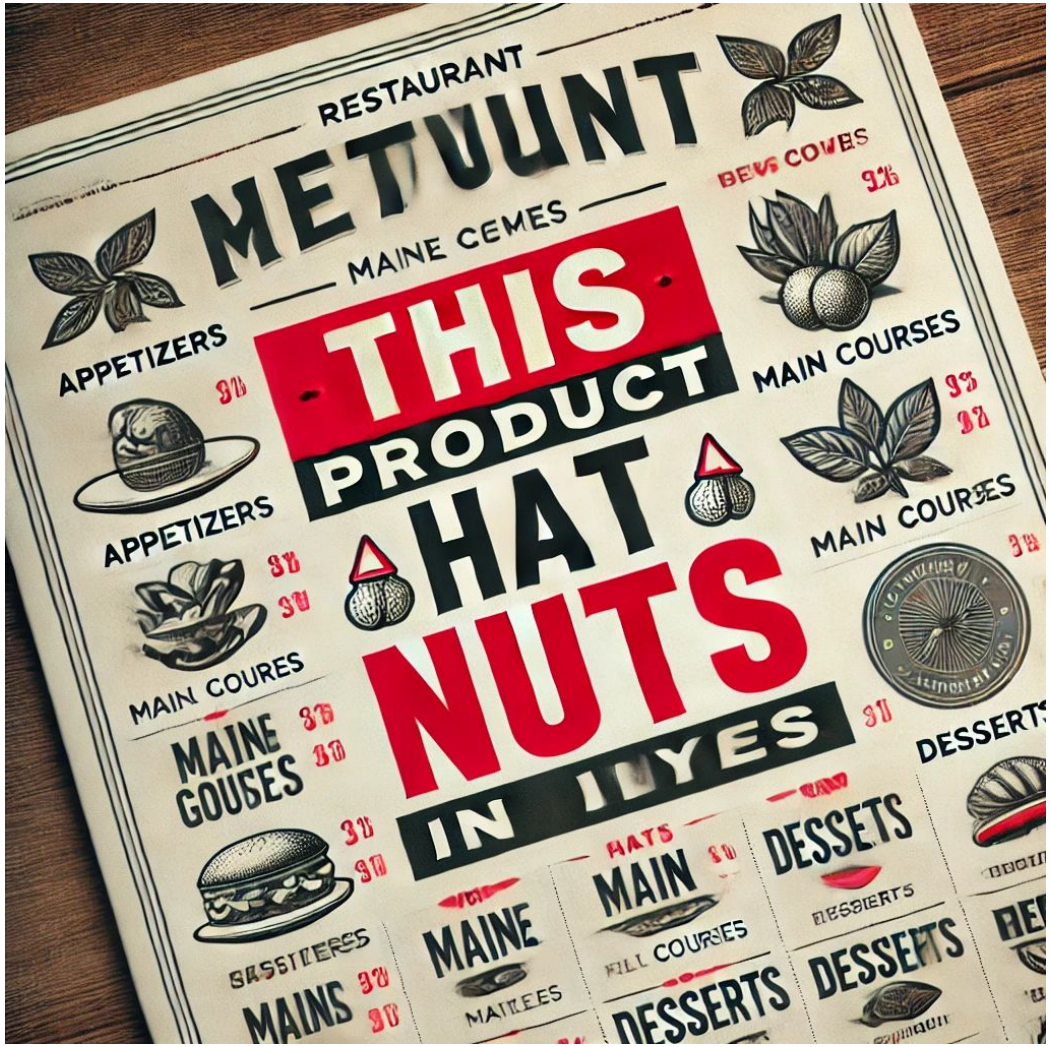
SBOM

SBOM – What is it

- List of software components
- An abstract map of the software dependencies

An ingredient list

Has it got nuts in it ?



Food labelling = risk management through a description of components (ingredients) and dependencies

nquiringminds

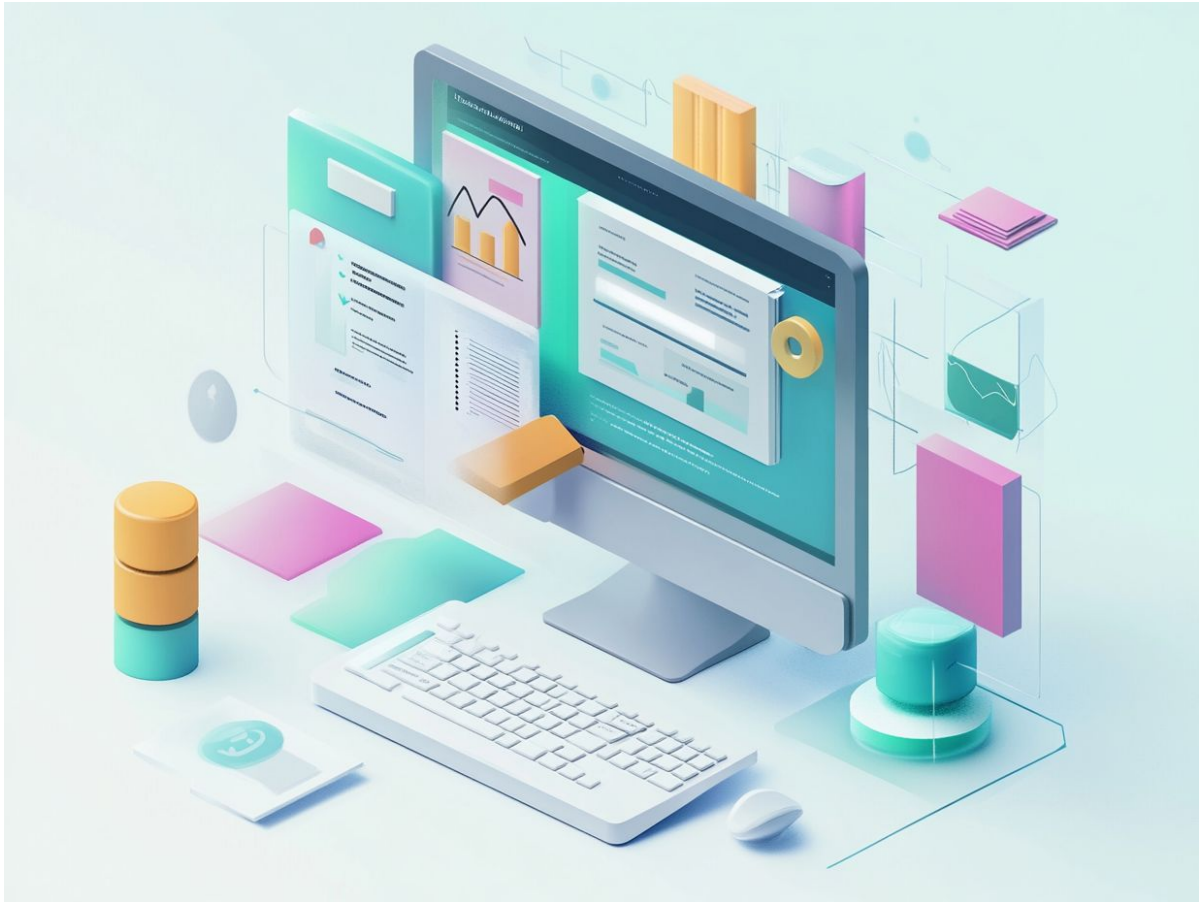
To contain, track and manage software supply chain risks

SBOM – How is it used

- List of software components
- An ingredient list
- An abstract map of the dependencies

Why…?

- To track the risks across the system (supply chain)



SBOM - Use cases

nquiringminds

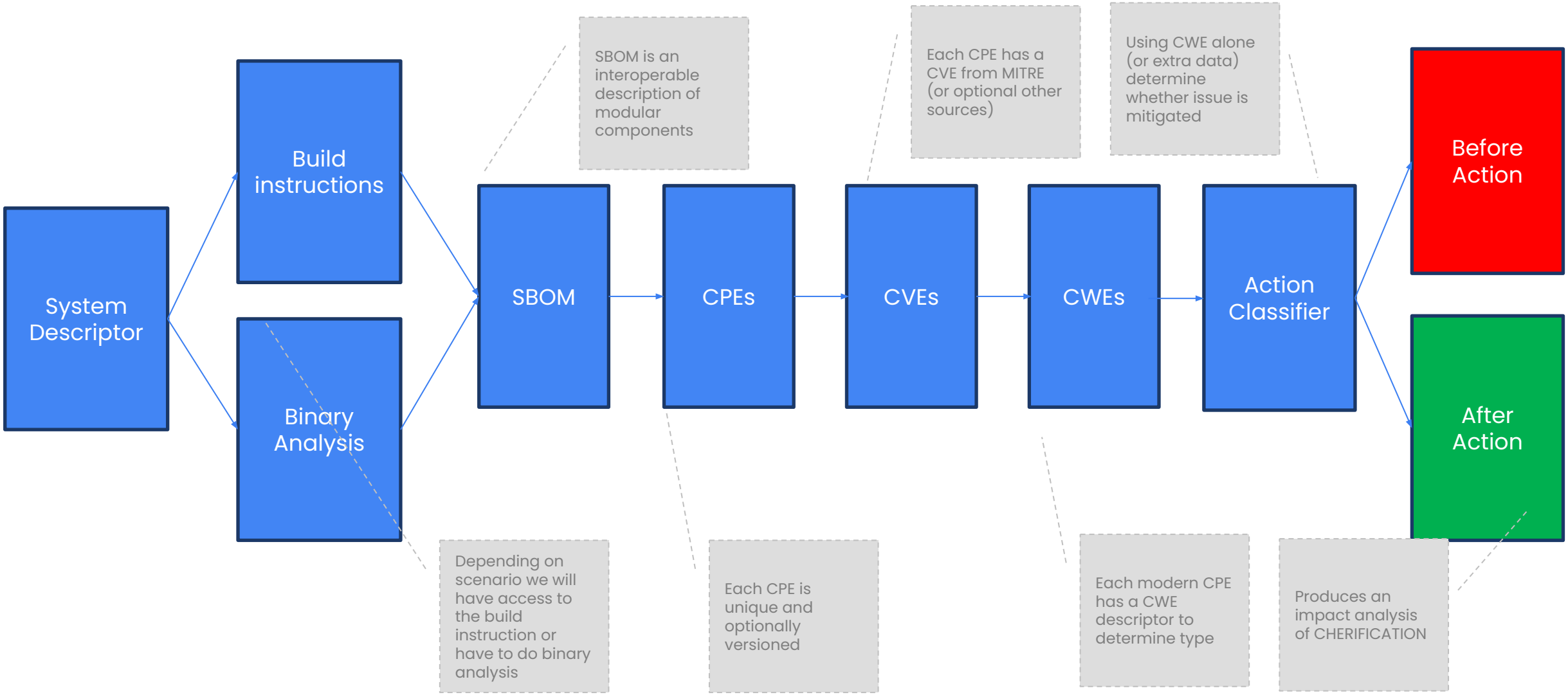
Risk type	Use case
Software vulnerability risk	Does my system have any critical vulnerabilities? A new critical CVE is announced in component X - which of my systems are impacted?
Export risk	Does my inventory contain any Foreign Ownership, Control, or Influence (FOCI) issues?
Licensing risk	Does my inventory contain any licensing risks - e.g. GPL pollution ?
Support risk	Under CSA (or other) regulations, what software support liabilities exist through dependencies on external (open source?) systems

Stakeholders

- **Developer**
- **Publisher**
- **Purchasing**
- **Operations**

nquiringminds

CHERI Impact Analysis

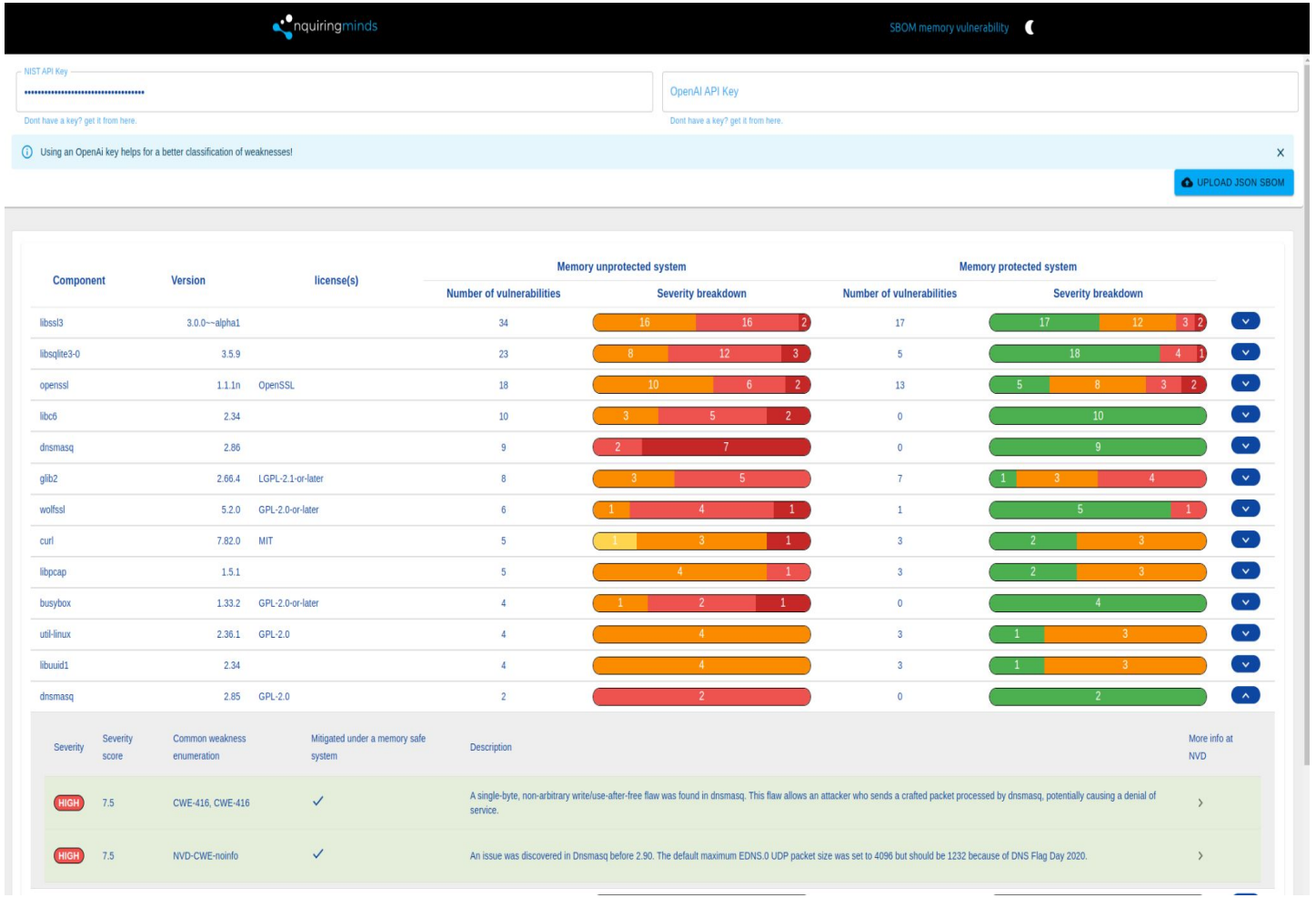


SBOM First Release

nquiringminds

Web APP

1. Comprehensive overview
2. Memory vulnerability Insights
3. Severity breakdown
4. Detailed weaknesses enumeration
5. Mitigation status



SBOM cyber sub-use case

nquiringminds

Integrated SBOM
Business case: evidencing value of an interventions
Development: evidencing and prioritising the code/system intervention to protect
Procurement: evidencing MemSafe system is better. Just buy the good stuff
Onboarding: selectively onboarding trusted MemSafe system
Impact analysis: when a new vulnerability is disclosed, identifying the impact surface (subsystems impacted)
Mitigation: addressing the impacted system (reduce cost)

Insurance: pricing cost of risk

nquiringminds

Vulnerability surface

Vulnerability surface (plan)

nquiringminds

Attack surface (defn): The attack surface of a software environment is the sum of the different points (for "attack vectors") where an unauthorized user (the "attacker") can try to enter data to, extract data, control a device or critical software in an environment.[1][2] Keeping the attack surface as small as possible is a basic security measure

Vulnerability surface (proposal): A subset of the attack surface. Is enumerable: the superset of all vulnerabilities, in all identified components of the software

Vulnerability surface (working defns)

nquiringminds

Vulnerability surface - envelope: the superset of identified vulnerabilities that can be mapped to a specific piece of software, by deconstructing the component of that software and aggregating all dependent (mapped) vulnerabilities

Vulnerability surface – active: a subset of the envelope created by removing all vulnerabilities that are not considered actively exploitable. Can be created using VEX or other method. (https://cyclonedx.org/capabilities/vex/)

Vulnerability surface - predicted: an estimate of the like hood of a new vulnerability being discovered in a particular component, which can be aggregated for a complete piece of software

An array of complex structures that can be mapped to a single number for comparison (sevirity, expoitabiltiy etc)

Vulnerability surface (next steps)

- An open source tool https://github.com/nqminds/SBOM-GAP
- SBOM-GAP provides a method to estimate
  - Vulnerability surface envelope
  - Vulnerability surface predicted
- Vadims paper:

And the other use cases

Export risk	Government and corporate risk. Legislation infringement. FOCI and national level threat
Licensing risk	Corporate risk and licensing compatibility
Support risk	Internal estimate: what is it going to cost me to to support my software dependencies under CSA regulations Invert: how valuable (as open source or proprietary provider) is the support I am providing

nquiringminds

SBOM Summary

Clear use cases - practical application

US/EU have strong poistion - UK ???

Imperfect - but fixable

For export alone intervention is essential

TAIBOM

To contain, track and manage information supply chain risks

nquiringminds

AI = Really complex software

> huge data dependencies

nquiringminds

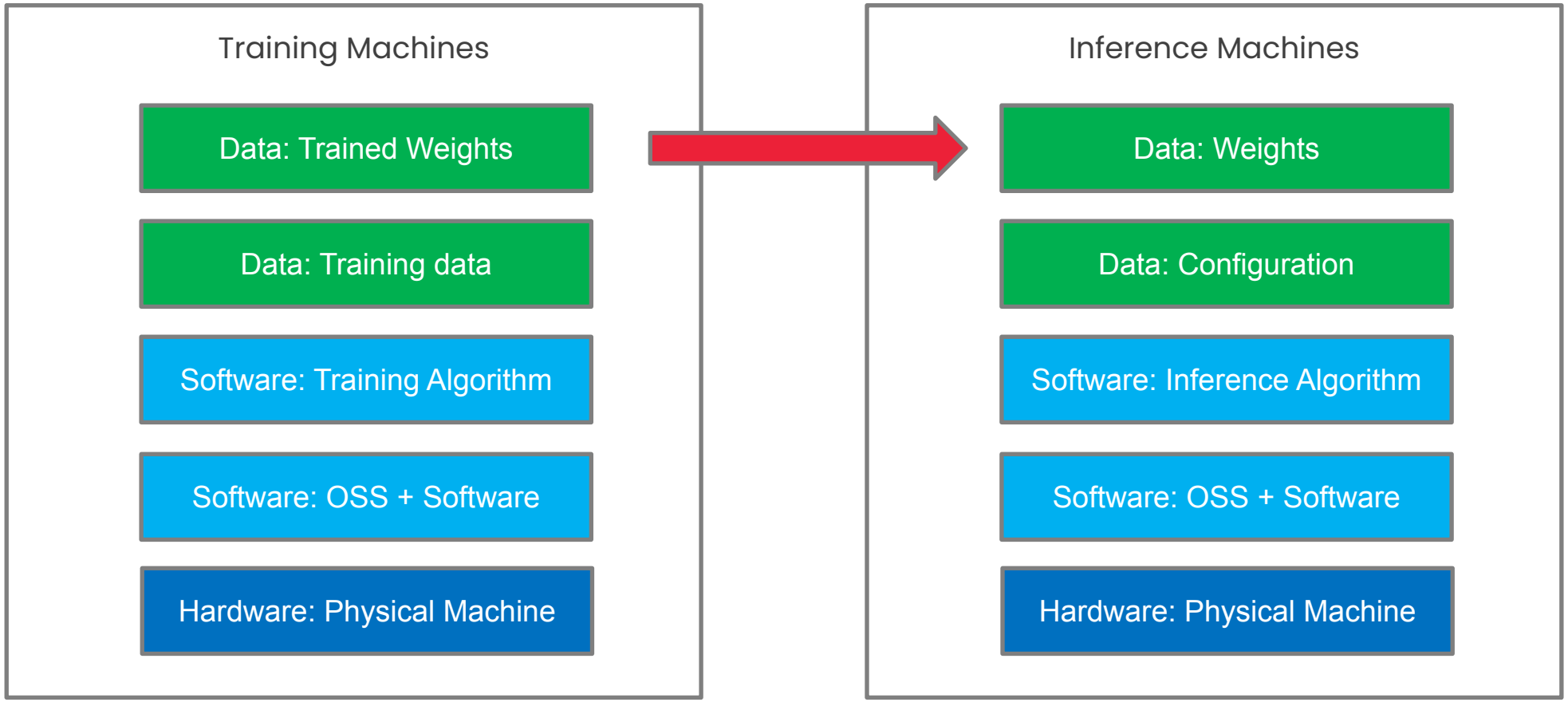
SBOM = risk in software component

AIBOM = risk in data component

nquiringminds

Simplified NN lifecycle

Most modern AI



AI System Challenges

Stability: AIs are complex dynamic data drive systems. Has it changed or been updated?

Dimensionality: The system is compromised if there is a compromise to any of

- Inference software
- Training software
- Trained weights
- Training data

Air gapped: The inference system is usually not connected to the training system

Distributed: a complex set of disconnected / un-related stakeholders

- Owner of data (separate to)
- Trainer of the system (separate to)
- Application developer (separate to)

Design features

nquiringminds

Label/version: foundational capability to label components

Dependencies: define risk dependencies between system and capabilities

Attestations: make subjective, extensible but interoperable assertions about components and systems

Distributed: no central point of control, works with air gapped system

Subjective: different between assertion and belief

Queryable : machine readable and interpretable at scale

TAIBOM OBJECTIVES

nquiringminds

Version/Label Components

What version of software is the inference/training system?

What version of the trained weights am I using?

What version of data was I trained on?

Describe dependencies

Inference system **depends on** inference SBOM + trained weights

Trained weights **depends on** training SBOM + training data

…. More complex….

TAIBOM **is not**

- A complete and perfect description of an AI system

TAIBOM **is**

- A static label that can be applied to AI systems and components versions
- An approximate model, which describes conceptual dependencies
- A method of making “subjective” attestations about components
- A method of propagating statements across (air-gapped) systems

TAIBOM - Use cases (SBOM derived)

nquiringminds

Risk type	Use case	TAIBOM changes
Software vulnerability risk	Does my **Inference** OR **Training** system have any critical vulnerabilities? A new critical CVE is announced in component X - which of my systems are impacted?	Need to consider dependency between training and inference system
Export risk	Does my inventory contain any Foreign Ownership, Control, or Influence (FOCI) issues?	Different export license surrounding AI EU specific regulation restrict use
Licensing risk	Does my inventory contain any licensing risks - e.g. GPL pollution ?	See copyright risk later
Support risk	Under CSA (or other) regulations, what software support liabilities exist through dependencies on external (open source?) systems	Unexplored what implications CSA has for AI systems

TAIBOM - Use cases - additional

nquiringminds

Risk type	Use case
Data poisoning	Has my training data been intentionally poisoned - and can I trace impact through to all deployed inference systems
Data pollution	Has my training data been accidentally polluted?
Performance checks	Do I have evidence that the system has been validated (performs well enough) for the application
Copyright risk	Is there any inherent copyright infringement risk in the data on which the system has been trained

TAIBOM - Use cases - additional

nquiringminds

Risk type	Use case
Bias risk	Are there inherent biases in either the data on which the AI system has been trained or in the performance on the versioned inference system
System tampering risk	Has the software or the trained weights been tampered with
Best practice/Legislation	Do I have evidence that the system designers employed best practice in the development of the system
Supply chain risk	Do I trust all the actors involved in the creation of the system. FOCI checks.

nquiringminds

TAIBOM HOW IT WORKS

TAIBOM – Basic capabilities

Labelling/Versioning

Every aspect of a complex AI system needs labelling and versioning. (data, code and physical systems). Ideally there should be a method of attesting to the version. There can be various trust models to implement this

Dependencies

A complex AI system has dependencies that need describing to fully understand provenance. TAIBOM will provide an interoperable method of describing these dependencies

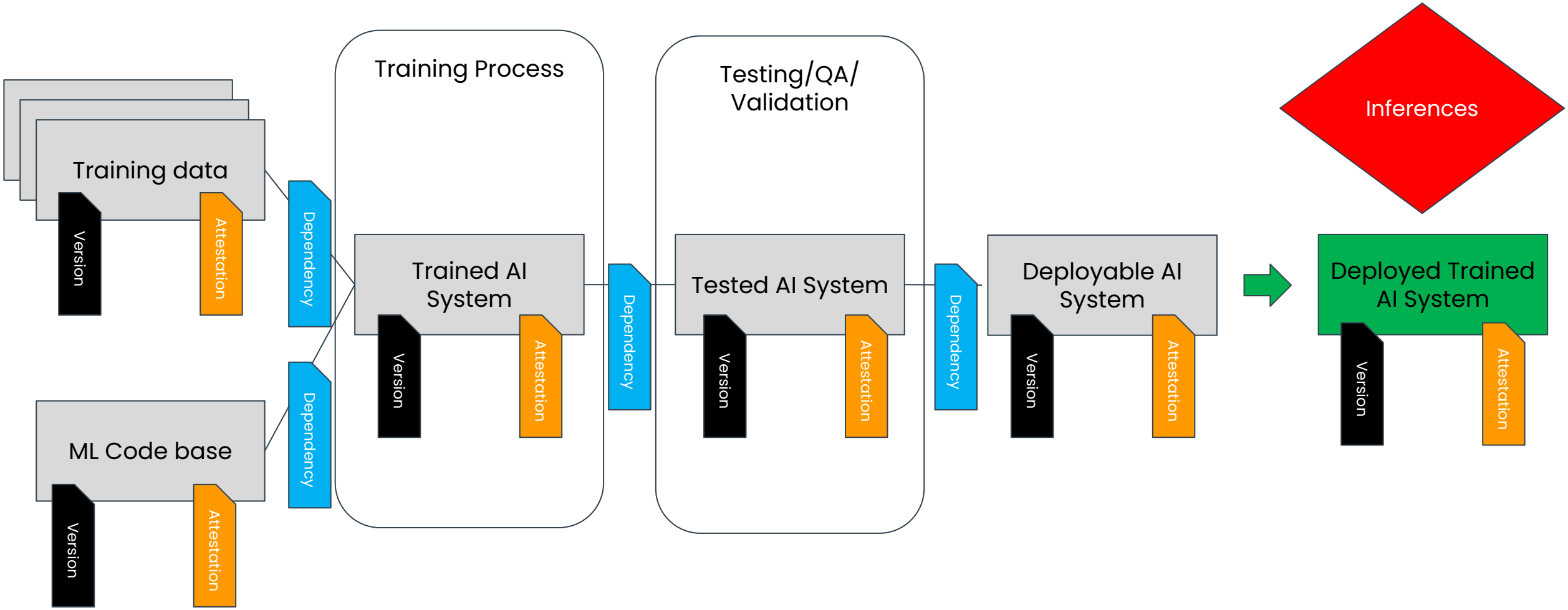
Attestation

Any actor (author or third party) can provide descriptors for each component of the system as a whole. (e.g. a training content review, as SBOM validation, a system integrity check, a fairness assessment).

TAIBOM provides both a mechanism of making these attestations, but also a framework for the dynamic and subjective evaluation, of combinations of these attestations.

All features fully decentralised

Claims and inferences



How TAIBOM works

nquiringminds

Label components

- Sign and version all training data sets
- Sign training code packages
- Sign SBOM descriptor and tie to training code
- Train system with configuration
- Sign trained weights
- Sign inference code
- Sign SBOM descriptor of SBOM code

W3C®

Verifiable Credentials Data Model v2.0

Sign with W3C VCs
Interoperable
Fully distributed

How TAIBOM works

nquiringminds

Describe dependencies

- Training data **depends on** all data sets
- Training system **depends on** training SBOM + training data
- Trained weights **depends on** training system + training config
- Inference system **depends on** inference SBOM + trained weights

Dependencies are counter signed VCs
Once system can reference another
With a label to describe the relations

How TAIBOM works

nquiringminds

Create attestations

- $Sign_x$(training-data, no-bias)
- $Sign_x$(training-data, poison-detected)
- $Sign_x$(training-data, best-practice)
- $Sign_x$(training-SBOM, no-vulnerabilities)
- $Sign_x$(inference-system, performance-good)
- $Sign_x$(inference-SBOM, no-vulnerabilities)

- Every attestation is another countersigned VC
- Anyone can countersign: developer, producer, auditor or users
- Attestation are extensible - use any definition of bias you like..
- System is fully distributed

How TAIBOM works

nquiringminds

Query the system

- Has my inference system been tampered with?
- Had the trained system declared its training data?
- Have licenses been acquired for training data?
- Has the training data been tested for poisoning?
- Was best practice employed in curation of training data?
- Are there known CVEs on the training system?
- Are there known CVEs on the inference system?
- Is the inference system performance above the domain specified threshold?

- Evidence is assimilated across the dependencies
- Evidence can be “Minimally disclosed”
- Queries formally run across the sum of the evidence
- Quires are subjective: do you trust the provider of the information
- Queries are dynamic: data can change, trust can change.
- Everything and forensic robust chain of evidence

nquiringminds

TAIBOM Summary

TAIBOM is SBOM ++

Labelling and versioning is **foundational**

Without labelling and versioning you can't say anything

It's a decentralized problem - needs decentralised primitives

Summary

nquiringminds

SBOM

It's not perfect - but it does provide value

EU and US have a taken a clear technical and political position

UK position is vague

Trade: SBOM impacts trade. UK companies need to comply with EU and US legislation. Intervention is needed to upskill

Security : There are operational and strategic security benefits to adopting SBOM processes in procurement, networking and other. We need guidance

Technical: SBOM is imperfect. Investment is needed to plug the holes

nquiringminds

TAIBOM

An open collaborative initiative

Builds on preexisting interoperable standards (W3C VCs)

Builds on emerging security practices (SBOM, CVE, model cards)

Reflects the distributed reality of AI systems (not a one stop shop)

Agnostic: does not mandate specific notions of performance or quality

Can propagate positive evidence: use of best practices

Can propagate negative evidence: vulnerabilities and poisoning

Extensible: simple cryptographic description that can be evolved over time

Get involved: https://www.techworks.org.uk/ai#el-c655c98d

Dr Nicholas Allott
nick@nqminds.com

BACKUP

TAIBOM - How used

Inventory

Generating An Inventory Of An AI System

A business acquires a competitor and inherits AI-related software assets. The management team of the acquiring business wants to understand the state of the assets, what their dependencies are, whether they have known vulnerabilities and what licences are involved. The business uses a TAIBOM client to scan the AI-related software assets to identify their component parts and then to search for relevant attestations.

This inventory can then be used to perform the typical checks, used in an SBOM scenario, for example:

- Check for vulnerability: the individual AI system components (and the system as a whole) can be checked against known vulnerability lists. In the case of software and host OS this can use systems such as CVE databases and GitHub vulnerabilities. In the case of data, we envision new databases being created that annotate data vulnerabilities or checks.
- Check for license compatibility: the individual AI system components (and the system as a whole) can be checked for license compatibilities.

TAIBOM – How used

Model Download

Downloading An AI Model From A Repository

A business is developing software that integrates a pre-trained AI model for recognising road signs. Domain-specific regulations require third party assessments of the robustness of the model to adversarial modification of road signs in the wild and internal cybersecurity teams require information about known vulnerabilities in the model and software assets associated with performing inference using the model.

The download of the model from the provider includes a TAIBOM with a hash of the model weights, a hash of the inference code and information about the training data, which is composed of multiple datasets.

After download, the downloaded weights and inference code are hashed by a TAIBOM client and compared to the respective hashes in the downloaded TAIBOM, confirming that the TAIBOM relates to the downloaded assets. The TAIBOM client searches for attestations relating to the TAIBOM according to the requirements of the downloader.

The search reveals attestations by a third party that the combinations of the downloaded weights and inference code meet the regulator's requirements, but also an attestation that one of the components of the training dataset is known to be poisoned.

TAIBOM – How used

Inference Case

Using Inferences Performed By A Third Party AI System

A business is integrating with an API-based AI service that is provided by a third party. The service searches the internet to provide answers to a user's questions. Before signing up to the service, the business downloads a TAIBOM that provides information about the underlying model, its training data and the data sources it can access during inference. Additionally, every inference API response is associated with a supplemental TAIBOM that provides the URIs of the websites that were used in the specific inference.

A search for attestations relating to the model reveals that it is vulnerable to a variety of prompt hijacking attacks, training data extraction attacks and prompt extraction attacks. The business decides to proceed with the service despite the attestations, but, for each inference, looks up the URIs in the supplemental TAIBOM to find attestations as to whether the pages represent a threat to the model and, if so, the result of the inference is subjected to automatic inspection.

To review and improve

What user cases are missing? How can we improve the description

Claim example summaries

Version Claims

Data
- Hashes
- URI

Software
- Hashes
- SBOMS (CPEs)

System Claims

Data
- Aggregation

Software
- Aggregation
- Dependency

AI
- Aggregation
- Dependency

Legal Claims

Data
- License

Software
- License

AI
- License

Claim example summaries

Data Behavioural

- Bias check
- Owner/author
- Poisoning check
- License conformance?

Software Behavioural

- Export conformance
- License conformance
- CVE/Vulnerability check
- Performance check
- Stability check

AI Behavioural

- Functional performance
- Bias validation
- Dependency scan
- Best practice conformance