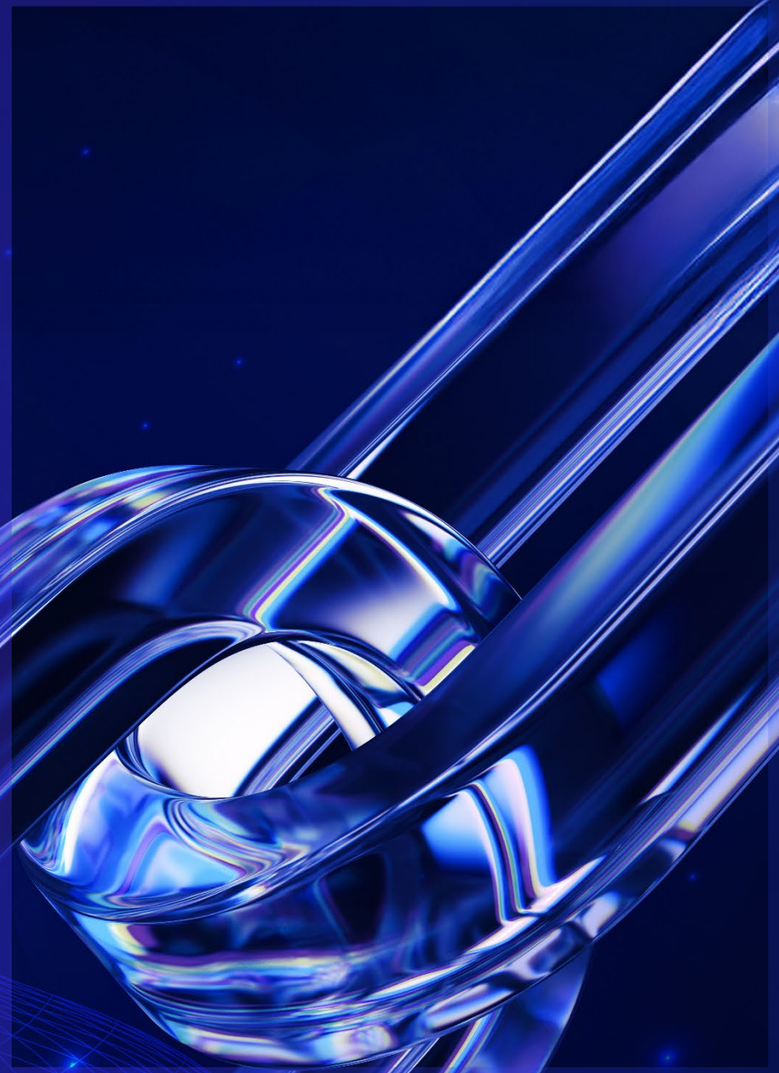
KPMG

LINEAJE

SBOM in India's Regulatory Landscape

Building Trust Through Transparency



December 2025

kpmg.com/in

KPMG. Make the Difference.

Foreword

Organisations' continued dependence on third-party software components, open-source libraries, and the increasing number of security incidents targeting these dependencies have helped Software Bill of Materials (SBOM) transition from a conceptual idea to a critical component of securing the software supply chain.

Indian regulatory bodies and agencies have started incorporating SBOM practices as a key component of their cybersecurity practices. From Computer Emergency Response Team (CERT-In's) comprehensive October 2024 and July 2025 guidelines to sector-specific requirements set by Securities and Exchange Board of India (SEBI) and Reserve Bank of India (RBI), it is evident that software transparency is now essential. This point of view (PoV) explores the importance of SBOM management in securing software products, and an approach to comply with regulatory requirements in India.

Understanding the regulatory requirements established by Indian authorities

In a significant initiative to fortify software supply chain security, CERT-In has enacted 'technical guidelines', first unveiled in October 2024, and subsequently expanded in July 2025. These guidelines mandate the incorporation of SBOM practices across key digital sectors. CERT-In's directives underscore India's strategic resolve in aligning with global cybersecurity frameworks, such as the U.S. Executive Order 14028, CISA, and the EU Cyber Resilience Act. By enforcing SBOMs, India is not only strengthening its internal cyber resilience but is also signaling its strong commitment to international software assurance and security standards.

Additionally, SEBI has integrated SBOM requirements into its Cyber Security and Cyber Resilience Framework (CSCRF), while the RBI has issued a circular in November 2024, compelling the inclusion of SBOMs for software products and components. This collective regulatory effort signifies a sector-specific enactment of CERT-In's broader vision, positioning India at the forefront of global cybersecurity innovation and leadership.

1. Indian regulators and agency key SBOM requirement summary:

SEBI[1]

1. SEBI mandates regulated entities to obtain and maintain SBOMs for existing critical systems, and for all new software procurements related to core and critical activities
2. Ensure that contracts with suppliers and third-party service providers include appropriate measures to meet the objectives of the regulated entity's cybersecurity programme and cybersecurity supply chain risk management plan
3. SBOMs should provide visibility for vulnerability tracking
4. SBOM must be updated with every software change
5. Document risk plans for legacy systems where SBOMs cannot be generated.

RBI[2]

1. RBI advises regulated entities (REs) to adopt CERT-In SBOM guidelines for enhanced software transparency and risk management
2. Generate and maintain SBOM for all software products and components
3. Share SBOM guidelines with internal teams and vendors, ensuring accuracy and timeliness
4. Integrate SBOM requirements into procurement and vendor contracts for compliance
5. Use internal assessments and audits to guide and review implementation
6. Maintain a CBOM[5] to catalogue cryptographic assets for quantum-safe preparedness.

Ministry of Power[3]

1. Power sector entities must reduce cyber supply chain risks by securing software and hardware procurement and deployment
2. ICT[6] procurement should use 'Trusted Sources and 'Trusted Products', or products must undergo malware and hardware trojan testing
3. All stakeholders, including OEMs[7] and vendors, must comply with the Ministry of Power's cybersecurity guidelines
4. Guidelines require vulnerability identification and component-level scrutiny, aligning with SBOM principles.

CERT-In[4]

1. Maintain an SBOM for every software asset across specified sectors
2. SBOMs should include all software components for full lifecycle traceability
3. Use standard formats like SPDX[8], CycloneDX, or SWID[9] for SBOMs
4. Continuously update SBOMs throughout the SDLC, using automated tools
5. Integrate SBOMs into security processes for quick threat identification
6. Retain SBOMs for compliance and audit readiness
7. Implement a phased SBOM adoption strategy for supply chain transparency
8. Define roles for SBOM management and ensure accountability
9. Secure and classify SBOM distribution with access controls
10. Establish a governance framework for SBOM management.

2. Entity Coverage:

1 **Software consumer:** Organisations that acquire software applications to support their operations, enhance productivity, and achieve their business objectives

2 **Software developer:** Organisations that develop customised software solutions

3 **System integrator/software reseller:** Organisations that distribute the software products and provide value-added services including customisation, integration, support, and training.

3. Software product type coverage:

Use cases

1 Source code developed and managed internally by organisation

2 Source code or software application developed by third party and managed/owned by organisation

3 Source code or software application developed and managed by third party.

1. **SEBI**: Securities and Exchange Board of India. *Cybersecurity and Cyber Resilience Framework (CSCRF) for SEBI Regulated Entities (REs)*, Circular No. SEBI/HO/ITD-1/ITD_CSC_EXT/P/CIR/2024/113, August, 2024.
2. **RBI: Reserve Bank of India,** Advisory No. 11/2024, Technical guidelines on Software Bill of Materials(SBOM) issued by CERT-In, November ,2024
3. **Ministry of Power**: Central Electricity Authority, Ministry of Power. *Draft Central Electricity Authority (Cyber Security in Power Sector) Regulations, October, 2021.*
4. **CERT-In**: Indian Computer Emergency Response Team, Ministry of Electronics and Information Technology. *Technical Guidelines on SBOM, QBOM, CBOM, AIBOM and HBOM,* Version 2.0, July, 2025.
5. **CBOM:** Cryptographic Bill of Materials
6. **ICT:** Information and Communication Technology
7. **OEMs:** Original Equipment Manufacturers
8. **SPDX:** *Software Package Data Exchange*
9. **SWID:** Software Identification Tag

Overview of Bill of Materials

Parameters	HBOM[10]	SBOM	CBOM	QBOM[11]	AIBOM[12]
Primary focus	Physical hardware components	Software components and dependencies	Cryptographic assets and usage	Quantum-vulnerable cryptographic assets	AI/ML models and datasets
Primary purpose	Validate hardware integrity and detect tampering	Identify vulnerable components, improve traceability, visibility of software components, and ensure licence compliance	Audit cryptographic configurations and enforce crypto hygiene, ensuring visibility into cryptographic posture	Prepare for post-quantum transition, plan migration to quantum-safe algorithms	Ensure fairness, traceability, and robustness in AI systems
Scope of inventory (Not limited to)	**Physical components:** processors, memory devices, circuit boards, communication interfaces, power components, sensors **Firmware and embedded software:** boot firmware, embedded OS, application firmware, security elements	Open-source libraries, proprietary code, dependencies, provenance, code quality, security posture, supplier and licenses, vulnerabilities, operating system packages, container images (docker layers and base images), scripts and configuration files, APIs and SDKs, plugins, and extensions (e.g., browser add-ons, IDE plugins)	Encryption algorithms, protocols, certificates, key stores and key management systems, hash functions, crypto libraries, usage context (data at rest/in transit), expiry and rotation policies	Quantum-vulnerable algorithms, key types and lengths, digital signature schemes, PKI infrastructure, crypto agility indicators, migration readiness status, quantum-safe zones and gaps, protocol using vulnerable algorithms	Training datasets, model architectures, pre-trained models and fine-tuned versions, feature engineering pipelines, ML libraries, hyperparameters and tuning configs, model lineage and versioning, inference environments, data sources and provenance, bias, and representativeness metrics
Key risk addressed	Prevents counterfeit hardware, backdoors, and tampering by ensuring traceability and integrity across physical components	Mitigates known vulnerabilities (CVEs), license compliance issues, and software supply chain risks by offering deep visibility into software components, dependencies, and provenance	Addresses weak cryptographic implementations, misconfigurations, and non-compliance with crypto standards such as FIPS/NIST	Identifies quantum-vulnerable algorithms and prepares systems for secure migration to post-quantum cryptography	Detects bias, adversarial ML threats, and IP leakage while ensuring AI systems are secure, fair, and explainable
BOM attributes (Not limited to)	Vendor details, part numbers, serial number, firmware reference, versions, country of origin, certification, data date codes	Component names, versions, supplier name, package URL, component dependency tree, license details, package checksum (SHA256), unique identifiers, vulnerabilities, VEX[13] document/section, security posture, code quality	Algorithm names, key lengths, protocol usage, certificate details, algorithm version, source library, compliance status, vulnerabilities	List of quantum-vulnerable algorithms and their locations, algorithm type, algorithm version, source library, vulnerability classification	Model versions, training data sources, libraries, audit trails and change logs, compliance status
Leading practices and regulations	CISA HBOM Framework (2023), HBOM Framework for Supply Chain Risk Management, CERT-In BOM Guidelines v2.0	U.S. Executive Order 14028, NTIA Minimum Elements for SBOM, NIST SP 800-218, EU Cyber Resilience Act (CRA), CERT-In BOM Guidelines v2.0, SEBI Cybersecurity Framework, RBI	NIST FIPS 140-3, SP 800-57, ISO/IEC 19790, CERT-In BOM Guidelines v2.0, RBI	NIST PQC Standards, ETSI Quantum-Safe Standards, CERT-In BOM Guidelines v2.0	OWASP AIBOM Framework (2025), EU AI Act, CERT-In BOM Guidelines v2.0,

As organisations increasingly recognise the importance of transparency in their software supply chains, the implementation of Software Bill of Materials (SBOM) programmes becomes crucial not only for securing software products but also for meeting regulatory requirements. From here on, our focus will shift towards understanding the SBOM adoption approach, common challenges associated with SBOM programme implementations, and exploring effective strategies to address them.

10. **HBOM:** Hardware Bill of Materials
11. **QBOM:** Quantum Bill of Materials
12. **AIBOM:** Artificial Intelligence Bill of Materials
13. **VEX:** Vulnerability Exploitability eXchange

SBOM programme adoption approach

While Indian regulations currently mandate the creation and maintenance of SBOMs for specific sectors such as banking, NBFCs, and power, organisations should go beyond compliance by establishing a robust SBOM management framework—one that progresses through preparedness (defining policies and tools), generation (creating SBOMs), monitoring (continuous updates and analysis), and integration (embedding SBOMs into security workflows) to build a mature and resilient cyber security capability. Outlined below is a phased approach that organisations may adopt:

Review and design

1. Identify and select in-scope software
2. Identify stakeholders responsible for different software lifecycle phases for each in-scope software
3. Identify and document the change management requirements to ensure stakeholders are onboarded for the new SBOM processes
4. Conduct readiness assessment and prepare a SBOM adoption roadmap
5. Design/uplift existing SBOM programme framework (including policy, SBOM format, minimum attributes, RACI matrix, and process workflows)
6. Develop business specification requirements documents for the selection and procurement of the SBOM platform.

Select and implement

1. Identify and evaluate the SBOM automation platform
2. Implement and configure the selected automation platform for all in-scope software products and components
3. Implement digital signatures or encryption to ensure the integrity and confidentiality of SBOM data
4. Use secure file-sharing platforms or tools that provide access control and audit capabilities
5. Establish a secure SBOM distribution channel.

Execute and operationalize

1. **Conduct training sessions** for relevant stakeholders (developers, security teams, and product managers etc.)
2. **For in-scope internal built software:**
   a) Generate and maintain SBOMs
   b) Analyse and validate the generated SBOM including VEX/CSAF[14] report
3. **For in-scope third-party and OSS[15] software:**
   a) Acquire/generate and analyse SBOMs as part of the procurement process
   b) Review and incorporate SBOM requirements for new third-party/OSS software contracts, and review and amend existing contracts to include SBOM clauses
   c) Obtain and validate SBOMs (including VEX/CSAF reports) for new and existing software products and generate internal SBOMs
   d) Review license types, usage and terms for OSS components and provide recommendations
4. **Risk assessment and remediation management:**
   a) Evaluate risks emerging from SBOM risk assessment outcome, including vulnerabilities, misconfigurations, and compliance gaps
   b) Formulate actionable remediation plans to address identified issues
   c) Apply patches, configuration changes, or other fixes to resolve identified issues, ensuring timely closure and documentation

Manage and monitor

1. Establish practices to securely store SBOMs (initially segregating individual SBOMs in dedicated repositories)
2. Periodic risk assessment and/or audit:
   a) Conduct regular audits of in-scope software products to validate SBOM completeness, accuracy, and alignment with compliance requirements
   b) Review SBOM generation, maintenance, and distribution processes to ensure consistency, traceability, and adherence to internal and regulatory standards
   c) Periodically assess risks associated with SBOM components of in-scope products, factoring in known vulnerabilities, dependency risks, and supply chain exposure
3. Provide SBOMs to customers, partners, and regulators as required, ensuring secure and controlled distribution.

SBOM programme governance

1. Establish an SBOM programme committee to oversee the SBOM programme execution
2. Conduct periodic SBOM programme governance meetings to evaluate and manage identified risks
3. SBOM programme KPI monitoring including SBOM coverage, vulnerability closure rate, and time-to-remediate
4. Review SBOM policies and processes annually for any significant changes.

14. **CSAF:** Common Security Advisory Framework
15. **OSS:** Open-Source Software

Common challenges faced by the organisations in establishing and implementing an SBOM programme (1/2)

Challenges	Description	Remediation steps
Source-code repositories visibility and mapping	Inconsistent repository mapping — such as when an application like *TravelApp* is linked only to its front-end repository while its back-end and shared library repositories remain unmapped — makes it difficult to identify all relevant codebases. This creates challenges in applying SBOM tools consistently and ensuring complete visibility of dependencies. The absence of standardised tagging - consistent labels for repositories, versions, and environments (e.g., frontend, backend, shared-lib, dev, stage, prod) — further hampers the ability to track which components belong to which builds. Without such tagging and clear environment separation, monitoring builds and ensuring complete dependency coverage becomes unreliable and error-prone.	1. **Centralised inventory:** Build and maintain a single source of truth for all repositories to clearly define application boundaries 2. **Dependency mapping:** Use specialised tools to automatically discover and visualise inter repository dependencies and connections 3. **Automation and tagging:** Implement automation that supports branches/tags, enforces standardised tagging conventions, and ensures traceable software lineage across builds and environments 4. **Ownership and governance:** Assign clear accountability for each repository, supported by standardised documentation to improve transparency and governance 5. **Monitoring and auditing:** Conduct regular audits of repository mappings and tagging practices to verify that all repositories are associated with their respective applications and environments. Leverage dashboards to surface unmapped repositories, missing tags, and inconsistencies, enabling corrective actions and ensuring end to end visibility.
Technology fragmentation	Diverse **application and technology architectures** — such as monolithic versus microservices designs, or teams using different frameworks like .NET, Gradle, and Maven - combined with **inconsistent naming of repositories, modules, and build artifacts**, hinder the automation and accuracy of SBOM generation. Integration with **CI/CD pipelines and SBOM tools** becomes complex because each framework and build system produces outputs in varying formats. This results in fragmented pipelines and significant challenges in achieving end-to-end visibility and traceability of software components.	1. **Standardise naming and coding practices:** Enforce consistent naming conventions for repositories, modules, and artifacts to reduce ambiguity 2. **Adopt scalable SBOM tools:** Use automation solutions that support multiple languages, frameworks, and build tools, integrating seamlessly with CI/CD pipelines 3. **Integrate into CI/CD pipelines:** Embed SBOM generation into build/release pipelines with standardized outputs across environments 4. **Continuous monitoring:** Periodically review SBOM outputs and naming conventions across different frameworks and build systems. Ensure SBOMs are complete, enforce standardised artifact naming, and identify gaps in CI/CD integration to preserve full traceability of software components.
Onboarding legacy applications	Onboarding legacy applications into an SBOM platform is complex because these systems often lack **core metadata attributes** such as version control history (e.g., Git commit IDs), repository references (e.g., GitHub or Bitbucket URLs), and component traceability (e.g., dependency manifests like **pom.xml** for Java or **package.json** for Node.js). Without these attributes, it becomes difficult to accurately identify software components, track their origins, or establish relationships between modules. **Example**, if a legacy Java application does not maintain its **pom.xml** file, SBOM tools cannot reliably capture its library dependencies. This gap undermines the ability to generate a **comprehensive and reliable SBOM**, leading to incomplete visibility of dependencies and potential security exposure.	1. **Identify components:** Use manual reviews and automated scanning to detect software components and fill metadata gaps 2. **Catalogue legacy applications:** Maintain a centralised registry with SBOM completeness status and supporting metadata 3. **Assess and migrate:** Evaluate feasibility of moving legacy apps to modern platforms, prioritising high-risk or compliance-critical systems 4. **Formalise risk acceptance:** Where SBOMs cannot be generated, document risk acceptance with compensating security measures 5. **Monitor and govern:** Periodically audit SBOM coverage, enforce governance, and provide visibility to stakeholders.

Common challenges faced by the organisations in establishing and implementing an SBOM programme (2/2)

Challenges	Description	Remediation steps
Open-source management	1. **Transitive dependency visibility:** Decentralised adoption of open source, inconsistent governance policies, fragmented tooling, and uncontrolled dependencies create significant visibility gaps. These gaps are most pronounced when tracking transitive components—indirect dependencies introduced by primary libraries. Limited visibility makes it difficult for organisations to maintain transparency across the software supply chain and to identify vulnerabilities early. **Example:** A project may directly include a popular JavaScript library, but that library could pull in dozens of transitive dependencies. Without proper tracking, outdated or hidden components may remain unnoticed, reducing visibility into potential risks. 2. **Unmaintained OSS component security:** While unmaintained OSS components may not necessarily cause visibility challenges, they pose a serious security risk. Vulnerabilities in such components often remain unpatched, leaving applications exposed to exploitation. This increases the likelihood of security breaches, licensing conflicts, and regulatory compliance failures. **Example:** If one of the transitive dependencies is outdated or unmaintained, vulnerabilities such as Log4j (CVE-2021-44228) can persist, exposing the application to severe security and compliance risks.	1. **Implement an SBOM automation platform**: Adopt a solution that generates SBOMs, identifies open-source components, tracks maintainers, highlights associated risks, and explicitly captures transitive dependencies. This helps ensure complete visibility of both direct and indirect components across the software supply chain 2. **Establish a remediation framework** for vulnerabilities: a) **Source and rebuild**: When a fixed version or patch is available for a vulnerable transitive component, update to the resolved version and rebuild the application with the full dependency hierarchy to maintain integrity and security b) **Code and rebuild:** If no fix or updated version exists, refactor or replace the affected component manually, then rebuild the application with the complete dependency tree to eliminate the vulnerability while preserving functionality 3. **Govern adoption:** Enforce clear OSS policies for licencing, approved repositories, and maintenance requirements through automation 4. **Monitor continuously:** Track vulnerabilities, license changes, and SBOM completeness with dashboards and audits to help ensure compliance and security.
SBOM disclosure reluctance	Vendors and distributors frequently resist SBOM disclosure due to concerns that exposing detailed component inventories could reveal **confidential design information, proprietary dependencies, or intellectual property**. This reluctance creates barriers to **supply chain transparency**, limiting downstream consumers' ability to validate vulnerabilities, licensing obligations, and compliance risks. **Example:** A hardware vendor may hesitate to share an SBOM for embedded firmware, fearing competitors could infer proprietary architecture choices or third-party licensing strategies.	1. **Contractual enforcement of SBOM requirements** a) Integrate SBOM disclosure obligations into **vendor contracts** during both new procurement and renewal cycles b) Specify the required SBOM format (e.g., SPDX, CycloneDX) and frequency of updates c) Include clauses for confidentiality protection to balance transparency with intellectual property concerns 2. **Leverage security assurance reports (VEX/CSAF):** Request **VEX** or **CSAF** reports to validate vendor security posture and map risks to SBOM data 3. **Establish controlled sharing mechanisms** a) Implement **secure portals or access-controlled repositories** where vendors can share SBOMs without exposing sensitive IP broadly b) Allow for **redacted SBOMs** (e.g., omitting proprietary modules) while still providing visibility into open-source and third-party components.

Case Study

Background

A leading Non-Banking Financial Company (NBFC) was facing challenges in meeting evolving regulatory and security requirements. With the Reserve Bank of India (RBI) mandating stricter compliance measures, the organization needed to generate and maintain Software Bills of Materials (SBOMs) for all its software applications—including legacy systems, third-party solutions, and open-source software (OSS) developed across more than twelve programming languages.

The primary objective was to achieve regulatory compliance while strengthening supply chain transparency and risk management. By implementing SBOMs across more than twenty critical applications, the client aimed to gain enhanced visibility into key risk domains such as component provenance, code quality, supplier trustworthiness, licensing obligations, and overall security posture.

In addition to SBOM management, the NBFC identified the need to **integrate its SBOM platform with a vulnerability management process.** By leveraging SBOM insights to identify existing vulnerabilities, the organization aimed to enable proactive remediation and ensure continuous risk reduction.

Approach:

To meet these objectives, organisation deployed Lineaje Inc's technology platform on-premises, facilitating SBOM generation and lifecycle management for critical applications across diverse technology stacks. SBOMs were created in the CycloneDX format for each release cycle, capturing relevant attributes in alignment with CERT-In practical guidance to evaluate and manage the risks associated with software components used in in-scope software applications. Utilising SBOM outcomes, we conducted risk assessments and provided recommendations to address identified vulnerabilities and issues. Alongside SBOM generation, we assisted in producing VEX and CSAF reports covering relevant vulnerabilities. By developing a centralised repository view, we facilitated the deployment of SBOM tools and provided an executive-level dashboard for strategic oversight.

Potential key benefits

1. Compliant SBOM for in-scope software product and applications
2. Strengthened alignment with SEBI and RBI regulatory requirements
3. Increased insight into supply chain risks associated with software components and dependencies in critical applications
4. Automated SBOM generation and maintenance
5. Built a scalable framework for extending SBOM coverage to non-critical applications
6. Integrating SBOM generation into existing workflows reduced manual effort
7. Enhanced software supply chain security through traceable and actionable SBOM information.

How can KPMG in India support?

Our team of in-house trained professionals brings deep expertise in the design, execution, and automation of Software Bill of Materials (SBOM) programme. We enable organisations to proactively manage risks across the entire lifecycle of software products and their dependencies. To enhance our capabilities, we have established strategic partnerships with leading technology providers, allowing us to integrate advanced platforms into our service offerings. This helps ensure our clients benefit from cutting-edge tools and methodologies aligned with global leading practices and standards.

As part of our alliance with Lineaje Inc., we leverage their technology platform to generate and maintain provenance and Software Bills of Materials (SBOMs) for software products and components. The Lineaje Inc. platform provides comprehensive verification and attestation of software integrity and authenticity, covering both direct and transitive dependencies down to the Nth level. It also assigns assurance scores to each component, enabling informed risk decisions across various metrics like security posture, code quality, vulnerabilities, etc.

Furthermore, the platform also offers near real-time scanning and alerting for newly discovered or existing vulnerabilities, significantly improving remediation timelines, and enhancing overall software resilience. KPMG in India along with Lineaje Inc., can help with the below areas in your SBOM programme adoption journey:

1. **SBOM programme maturity assessment**
   Assess the maturity of your SBOM management programme across governance, tooling, and process integration, benchmark against industry-leading practices, and support continuous improvement across the software lifecycle

2. **SBOM programme readiness assessment**
   Review existing SBOM management practices against applicable regulatory expectations (e.g., SEBI, RBI, CERT-In), identify areas of improvement, and develop a Plan of Action and Milestones (PAOandM) to address identified gaps

3. **SBOM framework design**
   Design SBOM management policies and procedures aligned with global standards to secure internal, third-party, and OSS components, supported by governance and automation

4. **Technology platform selection and implementation**
   Support in the deployment and implementation of technology platforms for software product and component provenance management:

   a) **Generate and maintain** provenance and SBOMs

   b) **Analyse SBOM:** Assess each SBOM for potential risks including vulnerabilities, code quality, security posture, provenance, etc.; compliance violations, and tamper ability of the supply chain, based on the defined SSCS framework

   c) **Securely publish and share SBOM:** Publish compliant and attested SBOMs for products and SKU. Further, generate and share SBOM, VEX, or CSAF and other deployment mitigations to enable secure deployment of applications by customers

5. **SBOM risk assessment**
   Analyse SBOMs for risks including vulnerabilities, code quality, provenance, and compliance violations; generate assurance scores per component

6. **Remediation management**
   Log, track, monitor, and close the identified vulnerabilities and issues in software components (internal, third-party, and open-source components) by working with identified stakeholders and partners

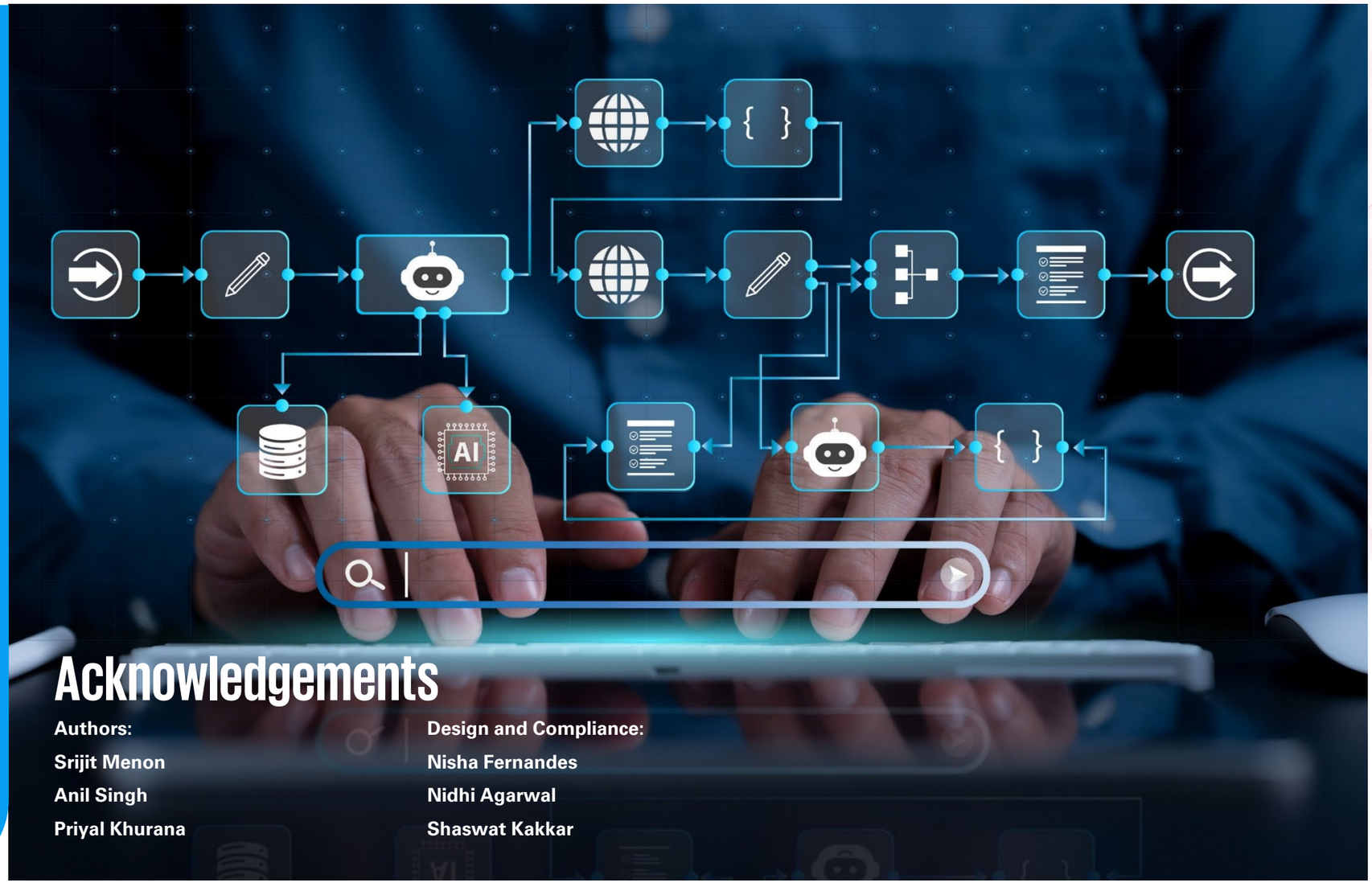
7. **Managed OSS**
   Establish governance and control over open-source software usage by integrating approved OSS components into SBOM workflows, continuously monitoring for risks, and helps ensure compliance with licensing and security standards.

Conclusion

Adopting a Software Bill of Materials (SBOM) is no longer a discretionary practice—it is a **strategic necessity** for organisations seeking to secure their software supply chains and meet both Indian and global regulatory mandates. While challenges such as incomplete metadata in legacy systems, fragmented open-source governance, limited repository visibility, diverse technology environments, and vendor reluctance to share SBOMs remain, the role of SBOMs in enabling **transparency, vulnerability management, and compliance assurance** is indispensable.

Success requires a well-defined **SBOM strategy** supported by robust processes for generation, validation, and lifecycle maintenance. Embedding SBOMs into cybersecurity frameworks with **continuous monitoring and automation platforms** accelerates adoption, while developing internal expertise helps ensures long-term resilience. Equally critical is **cross-functional collaboration** across engineering, security, risk, procurement, and business teams to align governance and strengthen trust across the digital ecosystem.

Looking forward, SBOMs are expected to expand into **extended BOMs (xBOMs)** encompassing AI models, SaaS platforms, crypto assets, and cloud services. Organisations that act decisively today are expected to not only achieve regulatory compliance but also establish themselves as **leaders in securing the integrity and resilience of today's digital supply chains**.



Acknowledgements

Authors:

Srijit Menon

Anil Singh

Priyal Khurana

Design and Compliance:

Nisha Fernandes

Nidhi Agarwal

Shaswat Kakkar

KPMG in India contacts:

Akhilesh Tuteja
Global Head – Cyber Security
E: atuteja@kpmg.com

Srinivas Potharaju
National Head - Digital Risk and Cyber, Partner KPMG
E: srinivasbp@kpmg.com

Anil Singh
Director
E: anilsingh1@kpmg.com

Atul Gupta
Partner and Head – Digital Trust
E: atulgupta@kpmg.com

Srijit Menon
National Head for TPRM in India
E: srijitmenon@kpmg.com

kpmg.com/in

Lineaje Contacts:

Javed Hasan
Co-founder and CEO
E: javed@lineaje.com

Anand Revashetti
Co-founder and CTO
E: anand@lineaje.com

Access our latest insights on KPMG Insights Edge

Follow us on:

kpmg.com/in/socialmedia