SECURITY & DEFENCE QUARTERLY

The software bill of materials as a defence capability: Supply-chain transparency, sovereignty, and alliance interoperability

Artur Chlebicki

contact@chlebicki.art

https://orcid.org/0009-0009-7304-0138

Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Starowiejska 62a/190, 80-534, Gdańsk, Poland

Abstract

This paper examines the Software Bill of Materials (SBOM), a machine-readable inventory of the components inside a piece of software, and asks what doctrine, procurement, and alliance policy must change for defence organisations to convert software transparency into an exercised military capability rather than a filed compliance artefact. The method is a conceptual analysis of the theory-adaptation type. A review of SBOM standards, mandates, and the emerging scholarly literature is combined with a structured reading of six software supply-chain incidents, selected to span distinct attack vectors and used as analytical lenses rather than as an empirical sample; one is read as a disconfirming case to test the argument's limits. The analysis identifies three defence functions that component-level transparency enables but the current acquisition leaves latent: (i) vulnerability triage at machine speed, (ii) supply-chain attribution and provenance, and (iii) allied interoperability and digital sovereignty. It sets out five barriers (classification, vendor resistance, legacy platforms, false confidence, and the mismatch between acquisition and software rhythms) and proposes a four-level capability–maturity ladder (filed, queryable, integrated, and federated), positioned against the existing constructs, such as supply-chain levels for software artifacts, cybersecurity capability maturity model, cybersecurity maturity model certification, and the National Institute of Standards and Technology's cybersecurity framework tiers. Defence procurement should require software composition data as a maintained, machine-readable capability requirement, enriched with exploitability context and exchanged between allies through format-neutral standards, so that reliance on foreign and open-source code becomes informed rather than be blind. The paper offers a North Atlantic Treaty Organization- and European Union-facing policy agenda and an agenda for measuring the operational value of supply-chain transparency.

Keywords:

cyber resilience, defence procurement, software supply chain, SBOM, NATO interoperability

Article info
Received: 4 July 2026
Revised: 10 September 2026
Accepted: 14 September 2026
Available online: 31 October 2026

Citation: Chlebicki, A. (2026) 'The Software Bill of Materials as a defence capability: Supply-chain transparency, sovereignty, and alliance interoperability', *Security and Defence Quarterly*, 56(4). doi: 10.35467/sdq/236757.

© 2026 A. Chlebicki. Published by War Studies University, Poland.
This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).

http://doi.org/10.35467/sdq/236757

Introduction

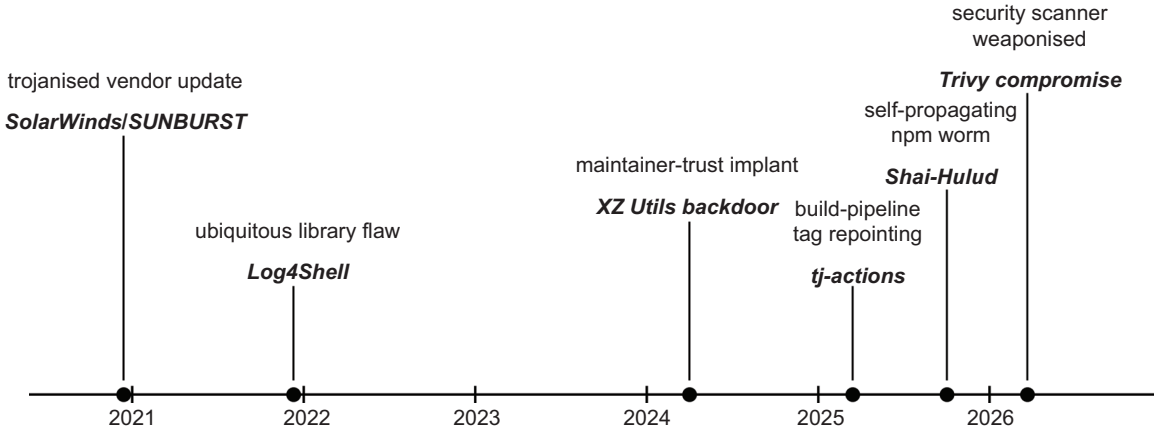
In late 2025, a piece of malware began to spread through the open-source software ecosystem on its own. Known as Shai-Hulud, it harvested the credentials of any developer who installed an infected package and then used them to inject itself into other packages that developer maintained, publishing the compromised versions automatically and with no human at the keyboard. Successive waves compromised hundreds of widely used packages faster than any manual response could follow (Microsoft, 2025). A few months later, in March 2026, attackers went one step further and compromised Trivy, one of the most widely used open-source tools for detecting exactly this kind of tampering, turning the scanner itself into a credential-harvesting weapon (Henig and Hyde, 2026).

These were not isolated events but the maturing of a method. A year earlier an actor had spent more than 2 years posing as a helpful volunteer to earn the right to commit code to XZ Utils, a data-compression library that runs unremarked on virtually every Linux system, before planting a backdoor that, routed through the *liblzma* dependency, hijacked remote login on any machine that shipped it; it was caught only by chance, by an engineer who noticed that secure-shell logins were running half a second slower than they should run normally (Freund, 2024). Whether patient and hand-crafted, like XZ Utils, or automated and self-propagating, like Shai-Hulud, each attack reached its targets through the same channel, the trusted software supply chain. Among those targets, unavoidably, are the defence networks assembled from the same open-source components. Figure 1 places these incidents, and the earlier compromises examined below, on a common timeline.

The lesson falls hardest on defence. A modern combat aircraft, air-defence battery, or command-and-control node is, in operational terms, a distributed computer whose lethality and availability depend on millions of lines of code, much of it open-source or commercial off-the-shelf, integrated through long and layered supplier chains that few programme offices can fully enumerate. The physical platform is visible and accounted for; the software it runs is, to a striking degree, not. This asymmetry between meticulous control of hardware provenance and near-blindness to software provenance is the problem addressed by this paper.

In the civilian world, the response has crystallised around a single instrument: the Software Bill of Materials (SBOM), a machine-readable inventory of the components, versions, and dependencies that make up a piece of software. Following the SolarWinds'

Figure 1. Timeline of the software supply-chain incidents used as analytical cases. Source: Author's own elaboration.



compromise, the United States made SBOMs a federal expectation through Executive Order (EO) 14028 (The White House, 2021), the European Union (EU, 2024) has embedded equivalent obligations in the Cyber Resilience Act, and European Union Agency for Cybersecurity (ENISA, 2025) has issued implementation guidance for operators of critical infrastructure. Yet within defence, where the consequences of an unseen dependency are most severe, SBOM is typically handled as a contractual deliverable: a document filed at acceptance, not a capability exercised in operations.

This paper argues that the compliance framing understates what software transparency offers armed forces, and asks: what would it mean to treat the SBOM as a defence capability rather than a procurement artefact, and what doctrine, acquisition policy, and alliance coordination would that require? The method by which that question is pursued, including the criteria by which the incidents above were selected, is set out in the next section. The contribution is threefold: it identifies three defence functions that software transparency enables (vulnerability triage at machine speed, supply-chain attribution, and alliance interoperability); it sets out the barriers that keep those functions latent; and it offers a North Atlantic Treaty Organization–European Union (NATO/EU)-facing policy and research agenda, organised around a maturity ladder that is positioned explicitly against the existing maturity constructs in this field. The contribution is conceptual rather than technical: SBOM tooling already exists; what is missing is the move that relocates software transparency from the compliance column of defence acquisition to the capability column.

Method

This is a conceptual paper of the theory-adaptation type in Jaakkola's (2020) typology: it takes a construct developed in software supply-chain security, puts it to work in defence capability development, and derives the functions, barriers, and policy implications that follow. The standards that apply are internal: the reasoning must be explicit, the material it operates is identifiable, and the claims no larger than the reasoning supports. The procedural choices are stated below, so the analysis can be re-run against a different incident set.

Case selection

Six software supply-chain incidents are used as analytical lenses, selected against the four criteria applied in the following order: vector distinctness, each case representing a compromise mechanism not already in the set, so that the six span the space of attack routes; documentation quality, meaning authoritative public sources (vendor incident reports, computer emergency response team (CERT) and Cybersecurity and Infrastructure Security Agency (CISA) advisories, review-board findings, and the national vulnerability database) rather than trade commentary alone; fleet-level consequence, since the argument concerns fleet-wide visibility and each case had to expose large numbers of downstream systems; and transferability to defence, each involving component types demonstrably present in military systems. Table 1 records the resulting set. Its span, from 2020 to 2026, covers a change of method, from patient hand-crafted subversion to automated self-propagation, that bears directly on the response-time argument.

Analytical steps

Each case was read in four steps: a description was assembled from the sources in Table 1; the description was put to the same counterfactual question at four phases separately

Table 1. The six incidents used as analytical lenses, with selection rationale. Source: Author's own elaboration.

Incident (year)	Attack vector represented	Read primarily for	Principal sources
SolarWinds/SUNBURST (2020)	Subversion of a vendor-built pipeline; malicious update, legitimately signed	Attribution and provenance; disconfirming case	FireEye (2020); SolarWinds Corporation (2020); US Government Accountability Office [GAO] (2021)
Log4Shell, Log4j (2021)	Critical flaw in a deeply transitive open-source dependency	Vulnerability triage at machine speed	Check Point Research (2021); Cybersecurity and Infrastructure Security Agency (CISA) (2021); US Cyber Safety Review Board (2022)
XZ Utils backdoor (2024)	Long-horizon subversion of a maintainer's position of trust	Attribution and provenance	CISA (2024); Freund (2024)
tj-actions/changed-files (2025)	Tampering with a mutable dependency inside a build pipeline	Limits of point-in-time inventories	National Institute of Standards and Technology (NIST) (2025)
Shai-Hulud Npm worm (2025)	Self-propagating, credential-driven package compromise	Speed asymmetry between attack and response	Microsoft (2025)
Trivy scanner compromise (2026)	Compromise of security tooling trusted to detect compromise	False confidence; tooling as a supply chain	Henig and Hyde (2026)

(with accurate, current, fleet-wide composition data, what changes at prevention, at detection, at response, and at attribution?); answers recurring across the majority of cases were promoted to the three functions set out below; and the question was then inverted, asking what would have had to be true, institutionally, for a favourable answer. The negative findings, clustered, gave the five barriers; and the enabling conditions, ordered by dependency, gave the maturity ladder.

Guarding against confirmation bias

A reading of this kind can select cases that flatter the instrument, and two devices work against that. Asking the counterfactual phase-by-phase forces the answer "nothing" wherever an SBOM would have changed nothing, which is the answer at the prevention phase for most of the set; and one case is read at length as a disconfirming case, because the compromise it represents would have been invisible to a perfectly accurate inventory. The resulting claims are narrower than the advocacy literature's: transformative for response and attribution, marginal for prevention, and dependent for much of their value on provenance controls the artefact itself does not supply.

The opaque stack: software-defined forces

The software content of military systems has grown faster than the institutions that procure them have adapted. Where a generation ago an avionics or fire-control function might have been realised in purpose-built, fully owned code, today it is assembled: programme offices integrate commercial off-the-shelf operating systems, middleware,

cryptographic libraries, machine-learning frameworks, and communications stacks, each of which in turn pulls in dozens or hundreds of transitive open-source dependencies. The resulting artefact behaves less like a single engineered object than like a supply chain frozen at a moment in time, and is only as trustworthy as its least-known link.

The scale of this dependence is well documented in the civilian sector. Black Duck's 2025 *Open Source Security and Risk Analysis*, drawing on 1,658 analyses of 965 commercial codebases, found open-source code in 97% of them and known open-source vulnerabilities in 86% of them. Most of the open-source components identified, 64%, were transitive dependencies, pulled in indirectly rather than chosen by the developer, and 90% of codebases contained components more than four years out of date (Black Duck Software, 2025). A defence system assembled from such components inherits both their vulnerabilities and their invisibility; there is no reason to expect military software, built on the same foundations, to be cleaner.

Two features make this stack genuinely opaque to its owner. First, dependency depth: a single top-level component may rest on a tree of sub-components several layers deep, most of which the prime contractor never selected and cannot name. Second, dependency drift: those components change continuously upstream, so an SBOM captured at acceptance describes a system that no longer exists after the first patch cycle. The consequence is a structural asymmetry. Defence acquisition applies exacting provenance controls to physical parts (every microchip, every alloy, and every fastener can in principle be traced) while accepting, for the software that actually determines what the platform does, a level of provenance ignorance that would be intolerable for any munition. This is the gap the SBOM is meant to close, and the reason its absence is an operational blind spot rather than a paperwork failure. A force cannot defend, patch, or attribute what it cannot see. Put in the language of command, a software-defined force lacks a recognised software picture: just as air and maritime operations depend on a continuously maintained common view of every track in the battlespace, forces that fight through software depend on a current, shared view of every component they field.

Three characteristics make the military case sharper than the civilian one. Defence platforms have service lives measured in decades, so the software they carry is routinely older than anything in a commercial data centre. Safety and airworthiness certification can freeze a configuration for years, turning "patch promptly" into an engineering and contractual impossibility just when a Log4Shell-class flaw demands speed. And the reassurance of physical isolation is partly illusory: an air-gapped system is still assembled from components delivered through a supply chain, so a compromise introduced upstream travels inside the perimeter regardless of how well that perimeter is guarded.

The SBOM: concept, standards, and mandates

An SBOM is, at its simplest, a formal, machine-readable record of the components that make up a piece of software, together with their versions, suppliers, and dependency relationships, the software analogue of the bill of materials long used in manufacturing. Its value is not the document itself but what the document makes possible: the ability to ask, across an entire fleet, "where is component X, and at what version?" and to receive an answer in minutes rather than weeks.

Two interoperable standards now dominate. Software package data exchange (SPDX), originally a Linux Foundation (2022) effort, was adopted as an international standard (ISO/IEC 5962:2021) in 2021 (International Organization for Standardization [ISO], 2021),

while CycloneDX, an Open Worldwide Application Security Project (OWASP) oriented towards security use cases, was ratified as an European Computer Manufacturers Association (ECMA) International standard (ECMA-424) in 2024 (OWASP Foundation, 2024). The coexistence of two formats matters for defence: it makes format-neutral tooling and explicit exchange agreements a precondition for any alliance-wide benefit rather than an afterthought.

The regulatory momentum behind SBOM is almost entirely civilian, and recent. In the United States, EO 14028, signed in May 2021 in the wake of the SolarWinds compromise (The White House, 2021), directed the production of minimum SBOM elements, published two months later by the National Telecommunications and Information Administration (NTIA, 2021) and carried forward through subsequent CISA (n.d.) guidance. In the EU, the Cyber Resilience Act, Regulation (EU) 2024/2847, makes the provision and maintenance of an SBOM a legal obligation for manufacturers of products with digital elements, requiring at minimum that it covers a product's top-level dependencies, with the relevant obligations applying from December 2027 (EU, 2024); ENISA (2025) has issued accompanying implementation guidance. Two further instruments complete the picture and matter particularly to defence readers. The network and information systems (NIS2) directive attaches duties to the operator rather than the manufacturer, obliging essential and important entities to manage supply-chain risk, including the security of relationships with direct suppliers and service providers (EU, 2022; Art. 21(2)(d)); and National Institute of Standards and Technology's (NIST, 2022a) cybersecurity supply-chain risk-management guidance supplies the control catalogue through which the US federal and defence buyers are expected to express such requirements contractually, which is the natural home for an SBOM clause. Defence, however, sits largely outside these instruments (national-security systems are commonly exempted), so the very sector with the highest consequences of failure has felt the least regulatory pull.

One further piece of the standards picture belongs here, because the argument below turns on it. An inventory records what is present; it says nothing about whether a given flaw in a listed component is actually exploitable in the product that carries it. That gap is filled by the Vulnerability Exploitability eXchange (VEX), whose minimum elements CISA (2023) has specified and which the common security advisory framework (CSAF) of the Organization for the Advancement of Structured Information Standards (OASIS, 2022) carries as a standard advisory profile. SBOM and VEX are designed to be used together, and defence acquisition that specifies the first without the second will find the first much less useful than expected.

A scholarly literature has also begun to form around the SBOM. This paper builds on it rather than beside it, so its present shape matters here. Its centre of gravity is software engineering. One strand maps the threat: Ohm *et al.* (2020) assembled the first systematic collection of real open-source supply-chain attacks, which Ladisa *et al.* (2023) developed into a taxonomy of more than a hundred distinct attack vectors. A second strand studies adoption and practice: Enck and Williams (2022) distilled the challenges reported by thirty industry and government organisations; Zahan *et al.* (2023) reviewed the benefits and obstacles of the post-EO 14028 mandate; and Xia *et al.* (2023) and Stalnaker *et al.* (2024) surveyed and interviewed practitioners, finding persistent uncertainty about what an SBOM should contain, how to keep it current, and what to do with it. A third strand is prescriptive: Okhravi *et al.* (2025) argue that the SBOM as mandated is essentially a reactive instrument, valuable only once an attack has been detected somewhere, and propose extensions (build pipeline description, reproducible builds, assurance and test metadata, and reputational attributes) that would allow it to support forward-looking trust assessments and so serve as a "proactive defence."

This paper accepts that diagnosis and pursues the question the literature has left open. Okhravi *et al.* (2025) ask what data the SBOM should carry; the question here is what an armed force must become in order to use it. The two answers bind together: the upper rungs of the maturity ladder proposed below presuppose exactly the build-generated, signed records that they call for, and their trust-assessment agenda in turn presupposes an institution able to act on the result. What is largely absent from the peer-reviewed record on either side is the defence case: the acquisition instruments, the classification constraints, the alliance-exchange problem, and the sovereignty stakes that distinguish a coalition of armed forces from a large enterprise. Those questions have so far been treated almost entirely in grey literature: vendor analyses frame the SBOM as foundational infrastructure for military and national security systems (Fortress Information Security, 2024), NATO's knowledge hub has examined controls against high-risk vendors (Kono and De Tomas Colatin, 2023), and the US Army has turned the case into procurement policy, requiring SBOMs for "covered computer software" (Bush, 2024). Where SBOM appears in defence contracts today, it therefore tends to arrive as an inherited compliance clause rather than a designed capability, and it is the gap that is addressed by this paper.

Reframing the SBOM as a defence capability

The argument of this paper is that software transparency, properly resourced, performs three functions that are recognisably military in character, each illustrated by one of the incidents in Table 1.

The word "capability" is doing specific work here, and defence planning gives it a precise sense. The North Atlantic Treaty Organization (NATO, 2026) defines a capability as "the ability to create an effect through employment of an integrated set of aspects categorised as doctrine, organisation, training, materiel, leadership development, personnel, facilities, and interoperability." The US joint practice applies the same logic through a slightly different set of lines of development, substituting policy for interoperability (DOTMLPF-P) (US Joint Chiefs of Staff, 2021); and in the capability-based planning literature, a capability is a set of resources and skills required to accomplish a task to a defined standard under defined conditions (Tagarev, 2009). Two consequences follow. First, materiel alone is never a capability: a delivered artefact without the doctrine that says who queries it and under what authority, the organisation and trained personnel to do the querying, and the interoperability agreements that let allies do it together, produces no effect. That is the condition of defence SBOM practice today, and it is why "we require SBOMs in our contracts" and "we hold this capability" are not the same statement. Second, capability is the currency in which defence planning transacts: the NATO (2026) defence planning process apportions targets to allies in capability terms, so framing software transparency as a capability claims a place for it among the requirements that drive investment rather than in the contract annexes that follow them. The three functions mentioned below should therefore be read as things a force becomes able to *do*; the bare inventory only enables them.

Vulnerability triage at machine speed

The defining lesson of Log4Shell (CVE-2021-44228), the maximum-severity (CVSS 10.0) vulnerability disclosed in the ubiquitous Apache Log4j 2 logging library in December 2021, was not the severity of the flaw but the chaos of the response. Because Log4j is a deeply embedded transitive dependency, most organisations could not quickly

answer the only question that mattered (*do we run it, and where?*) and spent days or weeks manually hunting through systems while adversaries scanned at scale; Check Point Research's (2021) telemetry recorded attempted exploitation against some 48% of corporate networks worldwide within days of disclosure. The urgency was such that the US Cybersecurity and Infrastructure Security Agency issued Emergency Directive 22-02 ordering federal agencies to find and remediate affected assets (CISA, 2021), and the US Cyber Safety Review Board (2022) later concluded that the difficulty of locating Log4j across estates made it an "endemic vulnerability" likely to persist for a decade. An organisation holding current SBOMs for its fleet could have converted that search into a database query. For armed forces, where the affected "systems" may be deployed platforms with narrow patching windows, the difference between minutes and weeks of exposure is a difference in readiness, not merely in IT hygiene. In capability terms, the gain is speed: response begins with a look-up rather than with a search.

An inventory on its own, however, over-reports. A fleet-wide query for a vulnerable component returns every instance of it, including the many in which the vulnerable code path is never reached, is not built into the shipped binary, or is already neutralised by configuration. In a large estate, this false-positive load lands on the scarce analyst capacity on which the whole capability depends, and a triage function that cries wolf will in practice be ignored. The complement is exploitability context: a VEX statement records whether a given vulnerability leaves a given product not affected, affected, fixed, or under investigation, with a justification required for the first (CISA, 2023), and may be carried inside the SBOM or distributed as a separate CSAF advisory (OASIS, 2022). The operational unit is therefore the SBOM, the VEX statements, and a machine-readable channel through which supplier assertions arrive and are updated. The inventory says what a force holds; the exploitability statement says which part of it matters now.

Supply-chain attribution and provenance

The SolarWinds/SUNBURST compromise (FireEye, 2020) demonstrated the strategic form of the threat: rather than attack a hardened target directly, an adversary subverted the build pipeline of a trusted vendor and distributed a backdoor through a legitimate and digitally signed software update. SolarWinds Corporation (2020) reported that fewer than 18,000 of its customers may have installed the trojanised Orion build, a figure later restated by the US government auditors (GAO, 2021), although the subset actively exploited was far smaller, a reminder that supply-chain compromise converts one breach into thousands of exposures. The XZ Utils backdoor (CVE-2024-3094) showed a second, patient variant, in which trust in a maintainer rather than a build system was the asset subverted (CISA, 2024); and in March 2025, the tj-actions/changed-files action, used across tens of thousands of automated pipelines, had its version tags quietly repointed to a malicious commit that leaked build secrets into publicly readable logs (NIST, 2025). None of these attacks targeted a perimeter; all targeted provenance. An SBOM does not prevent such compromises, but it is a precondition for detecting them: it establishes what *should* be present, against which anomalies and untrusted suppliers can be assessed, and it lets a defender ask, the moment a vendor or upstream library is implicated, "which of our systems carry this supplier's code, and at what version?," turning a vendor compromise into a bounded triage problem rather than an open-ended hunt. In a grey-zone contest where deniability is the adversary's principal asset, the ability to reconstruct what entered a system, from whom, and when, is the foundation of any credible attribution, which in the cyber domain remains a matter of judgement rather than proof (Rid and Buchanan, 2015), and of any proportionate response.

Alliance interoperability and digital sovereignty

The third function is collective. NATO operations depend on coalition systems sharing components, which means, they also share vulnerabilities and supplier risks. Without a common way to express and exchange software composition, each ally re-discovers the same exposure independently and at different speeds, the Log4Shell problem multiplied across a coalition. A shared, format-neutral SBOM practice would let allies propagate a "where is component X?" query across national fleets and coordinate a response; transparency then works as an interoperability asset. It also speaks to sovereignty: a state can only make sovereign decisions about acceptable supplier risk if it knows the provenance of the software in its own systems, rather than inheriting those decisions silently from an upstream it never chose, the concern now driving the EU's turn to digital sovereignty and strategic autonomy (Farrand *et al.*, 2024). NATO (2024) has begun to treat supply-chain security as an institutional concern: its 2024 defence-critical supply-chain security roadmap signals alliance-level attention, and NATO commentary identifies software supply-chain attacks of the Solar Winds type as a strategic risk to allied digital infrastructure (Luenen and Brooks, 2024). Yet this work has yet to translate into a shared and operational SBOM practice across members.

Sovereignty in this context is not autarky. No European state can or should build its defence software wholly from domestic components; the realistic aim is informed dependence rather than blind dependence. An SBOM is what makes dependence informed: it lets a government see which foreign or single-source components its forces rely on, judge which of those dependencies are tolerable, and which warrant a second source or a mitigation, and treat that as a deliberate policy choice rather than a discovery made in the middle of a crisis.

Where transparency helps least: a disconfirming case

The three functions are claims about particular phases of an incident. Reading one case against the grain shows where the instrument stops reaching. Take SolarWinds prospectively rather than retrospectively. Orion was a known, contracted, and legitimately licensed product; the trojanised build carried the same product name and the same version string as the genuine one and was signed with the vendor's own certificate. A perfectly accurate, continuously maintained SBOM would have recorded exactly what was in fact fielded, and would have flagged nothing, because the compromise lay in the build rather than in the component list. The same is true of the Trivy compromise in 2026 and, at the moment of installation, about the tj-actions tampering: an inventory that faithfully records a component cannot reveal that the component's binary is not what its source implies. What SBOM changed in those cases belongs to the phases after disclosure: scoping exposure, prioritising response, and bounding attribution. The claim is real, and much narrower than usually intended by the advocacy literature.

That has consequences for how the case should be put. Claims for software transparency belong at the phase where they hold, so that acquisition language does not encode the category error of buying an inventory and expecting prevention. Prevention, where it is achievable at all, needs the provenance layer the inventory does not itself supply: build attestation, reproducible builds, and signed artefacts. That is the extension Okhravi *et al.* (2025) argue should be folded into an SBOM, and the upper levels of the maturity ladder below treat it as the threshold of genuine capability rather than an optional refinement.

Barriers to adoption

If the case is this strong, why is SBOM still treated as a checkbox? Five barriers recur. First, the transparency paradox. The same component inventory that lets a defender locate a vulnerable library across a fleet in minutes would, in an adversary's hands, serve as a targeting map, a precise list of which exploit to load against which platform. Software transparency is in this sense dual-use: its defensive and offensive values rise together, so defence's reflex to classify composition data is not mere bureaucratic secrecy but a rational response to a real exposure. The task is to design handling regimes (tiered access, need-to-know distribution, and secure exchange) that capture the defender's advantage without handing the map to the attacker. Second, vendor resistance: contractors treat detailed composition as commercially sensitive intellectual property and resist disclosure absent a contractual requirement, which itself is an argument for specifying SBOM in acquisition. Third, legacy and embedded systems: much fielded military software predates SBOM tooling, runs on constrained hardware, or was delivered without the build-time instrumentation needed to generate an accurate inventory; so, retrofitting is expensive and sometimes infeasible. Fourth, and most insidious, false confidence: an SBOM is necessary but not sufficient, and a list of components that is incomplete, stale, or never cross-referenced against vulnerability and threat data offers the appearance of control while delivering little of its substance. The risk runs deeper still, because the tools relied on to provide that control are themselves part of the supply chain, as the compromise of the Trivy scanner described in the introduction illustrates (Henig and Hyde, 2026). Empirical studies of civilian SBOM practice already report this pattern of inconsistent generation, poor maintenance, and uncertainty about downstream use (Stalnaker *et al.*, 2024); defence, with its added classification and legacy burdens, has no reason to expect an easier path.

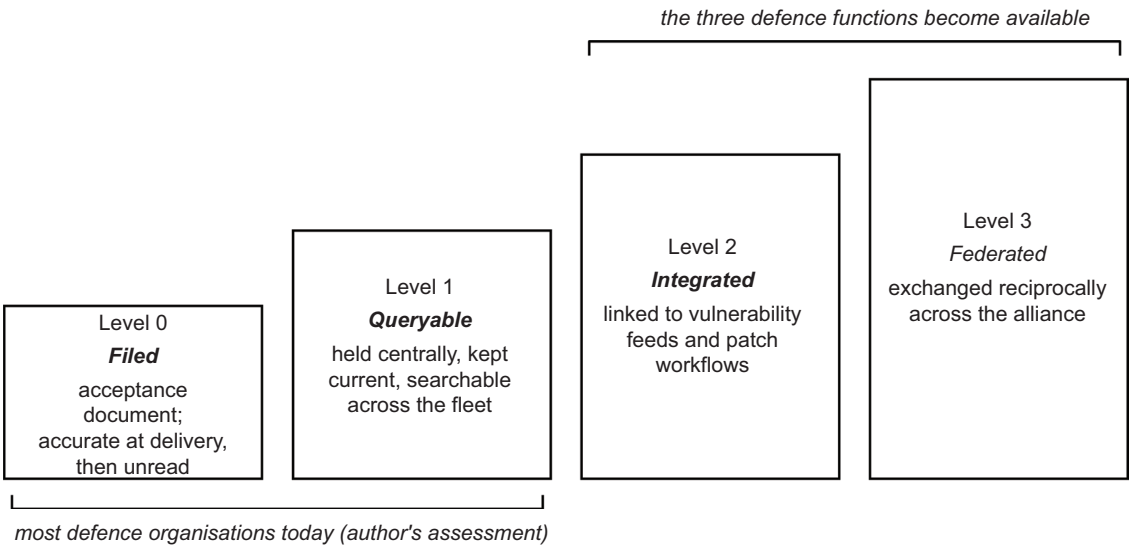
Fifth, and structurally, the rhythm of defence acquisition is mismatched with the rhythm of software. Programmes are scoped, contracted, and accepted over years, while the components they incorporate change weekly, so a procurement model organised around a single acceptance milestone struggles to accommodate an artefact that must be regenerated continuously to remain true. Until contracts treat the SBOM as a living deliverable carrying maintenance obligations, rather than a snapshot captured at delivery, even well-intentioned adoption will drift out of date almost as soon as it is in place.

A deeper objection cuts beneath all of these barriers. An SBOM, a sceptic argues, does not solve the trust problem so much as relocate it: a buyer who could not see the components must now trust whoever compiled the inventory. The objection is fair, and it sets a design requirement rather than defeating the case. An SBOM a supplier simply types out is worth little; one generated automatically from the build pipeline and signed cryptographically, so that its own provenance can be verified, is a different instrument altogether. That is why the maturity levels set out below treat automated generation and component signing as the threshold of genuine capability, and that is why the policy agenda that follows funds build-pipeline instrumentation rather than document templates.

A capability–maturity ladder for defence SBOM practice

A useful way to make this agenda concrete is to read SBOM adoption as a ladder of capability rather than a yes-or-no state of compliance. Four levels can be distinguished (Figure 2). At level zero, the SBOM is *filed*: it exists as an acceptance document, accurate on the day of delivery and consulted by no one afterwards. At level one, it is *queryable*: held

Figure 2. The SBOM capability–maturity ladder. Source: Author's own elaboration.



centrally, kept current, and searchable across the fleet, so that a force can answer "where is component X?" on demand. At level two, it is *integrated*: connected to vulnerability feeds, exploitability statements, threat intelligence, and patch-management workflows, so that a newly disclosed flaw triggers an automatic fleet-wide check rather than a manual campaign. At level three, it is *federated*: exchanged reciprocally with allies under the tiered regime set out below, so that coalition infrastructure carries shared visibility rather than shared blind spots. The three functions identified above become available only from level two upwards. The ladder's value is diagnostic: it tells an acquisition authority not merely to require an SBOM but where its practice stands and what the next step must be.

The ladder is a narrow instrument by design, and a defence organisation may be assessed against several existing maturity constructs at once. Table 2 states what each construct grades, whose practice it grades, and how it relates to the ladder proposed here.

The constructs complement each other rather than compete: an organisation certified at CMMC level 2, buying from suppliers at SLSA build level 3, may still sit at level zero of this ladder because nobody ever queries the data, and that combination is exactly what the diagnostic is meant to surface.

A word on where defence organisations actually sit on this ladder. The assessment offered here that most defence organisations are at level zero or at best level one is the author's judgement, inferred rather than measured, and should be read as such: no public dataset records defence SBOM operating practice, which is unsurprising, given the classification of the underlying material. The indirect evidence is consistent with it. The Linux Foundation's (2022) survey found production and consumption of SBOMs limited and inconsistent even among organisations that valued them; practitioner studies report generated SBOMs frequently stale, incomplete, and not acted upon (Stalnaker *et al.*, 2024; Xia *et al.*, 2023); ENISA (2025) describes an ecosystem still short of routine operational use; and the US Army policy that has gone furthest mandates delivery of SBOMs without the fleet-wide query infrastructure that level one would require (Bush, 2024). Establishing the actual distribution empirically is the first item on the research agenda mentioned below.

Table 2. The proposed ladder in relation to the existing maturity constructs. Source: Author's own elaboration.

Construct	What it grades	Whose practice	Relation to the proposed ladder
Supply-chain levels for software artifacts (SLSA) v1.0, Build Track L0–L3 (Open Source Security Foundation [OpenSSF], 2023)	Integrity and non-falsifiability of the build process and its provenance	Software producer	Supplies the trustworthy inputs that levels two and three presuppose
Cybersecurity capability maturity model (C2M2) v2.1, MIL0–MIL3 (US Department of Energy, 2022)	Maturity of a cybersecurity programme across ten domains	Operating organisation, programme-wide	Broader and organisation-wide; does not address fleet-wide composition query
Cybersecurity maturity model certification (CMMC) levels 1–3 (US Department of Defense, 2024)	Implementation of prescribed controls protecting federal contract information (FCI) and controlled unclassified information (CUI)	Defence supplier	Certifies a supplier's controls, not an operator's use of composition data
NIST CSF 2.0 Tiers 1–4 (NIST, 2024)	Rigour of cyber-risk governance and its integration into decisions	Organisation as a whole	Governanceposture; capability-agnostic
Filed/queryable/integrated/ federated (the present paper)	What operational question a force can answer about its own fielded software, how fast, and with whom	Operator of fielded systems, and the alliance	The unit of assessment is one capability, read from the consumer side

Policy implications for NATO and the EU

The reframing yields a practical agenda. First, defence acquisition authorities should specify SBOM as a *capability requirement*, accurate, machine-readable, and kept current across the system lifecycle and its patch cycles, accompanied by exploitability statements in a standard format rather than as a one-off acceptance deliverable. Second, allies should agree on format-neutral exchange (SPDX/CycloneDX interoperability) and shared tooling, so that composition data can cross national boundaries at the speed an incident demands; NATO's (2024) existing supply-chain work is a natural vehicle. Third, defence policy should align with, rather than exempt itself from, the civilian regulatory baseline set by the EU Cyber Resilience Act and NIS2 and the US federal practice, including the secure-development practices of NIST's (2022b) Secure software development framework and the supply-chain control catalogue of SP 800-161 (NIST, 2022a), adapting them to classification needs instead of starting from zero. Fourth, investment should target the unglamorous enablers, without which the inventory is inert: build-pipeline instrumentation, component provenance and signing, and the analyst capacity to act on SBOM data.

The paper's own contribution to this agenda is to extend the case of grey literature along a dimension it treats only nationally: a combined NATO–EU framing in which alliance interoperability and digital sovereignty are served by *reciprocal*, cross-coalition SBOM exchange, rather than by each state securing its own supply chain alone. Such exchange need not be all-or-nothing. It can be tiered: at the fullest level, allies furnish maintained SBOMs for the systems they contribute to shared or federated mission networks; at a minimal level (all that the Log4Shell case in fact requires), they expose only a queryable interface that answers "do your fielded systems contain component X at version Y?" without revealing rest of the inventory. That minimal tier substantially reduces the transparency

paradox set out above: it lets a coalition push an urgent component query across national fleets while each member discloses almost nothing about its wider composition.

It does not, however, eliminate the exposure, and it would be wrong to present it as information-free. A yes/no answer still discloses presence, which is itself targeting-relevant, and a sequence of such queries, made by a partner acting in good faith or by an adversary who has obtained access to the query channel, can reconstruct an inventory component by component or confirm a targeting hypothesis at low cost. The minimal tier therefore needs controls of its own: authenticated and logged queries, rate limiting, restriction of the queryable component set to items under active advisory, and reciprocity, so that querying is itself visible to the party being queried. Designing those controls, rather than assuming the minimal interface, is safe by construction, and is part of the policy work implied by this recommendation. These measures are modest relative to the platforms they protect, and they convert the existing compliance artefact into usable readiness.

Conclusions and research agenda

Military power now rests on software whose composition its owners largely cannot see, and, as the self-propagating supply-chain attacks of 2025 and 2026 make plain, that blindness is being actively exploited. The instrument designed to restore that visibility is being adopted in a framing that squanders its strategic value. This paper has argued that software transparency, treated as a defence capability in the sense defence planning gives the term, performs three military functions: vulnerability triage at machine speed, supply-chain attribution, and alliance interoperability and sovereignty. It has also been candid about the five barriers and about the phases of an incident at which an inventory changes nothing.

Three limitations bound these claims. The argument is conceptual rather than empirical, so the operational gains it describes are reasoned rather than measured. The incidents are chosen to illustrate distinct attack vectors and functions, not sampled to represent the threat as a whole. And the framing is deliberately NATO- and EU-centric, which leaves open how far the same logic carries to states outside those alliances.

The empirical agenda that follows has four items. First, and prerequisite to the rest, the distribution of defence organisations across the maturity ladder should be established empirically rather than inferred, through structured survey or interview work with programme offices and national cyber commands. Second, research should seek to *measure* the operational value of supply-chain transparency. One measurable starting point is the *mean time to locate* a named component across a fleet: the interval between a vulnerability's public disclosure and force's confident answer to "do we run it, and where?" For a blind force that interval runs to days or weeks, for an SBOM-mature one, it should approach the latency of a database query, and the distance between the two is a quantifiable readiness measure that acquisition policy could target directly. Third, how SBOM data can be integrated with attribution workflows in grey-zone conditions, and what evidential weight composition records can bear when attribution is contested. Fourth, how classification-compatible exchange mechanisms can be designed, so that allies share enough to act collectively without disclosing more than security allows, including the query-channel controls sketched above. The contribution here is to move the conversation from whether defence should hold a bill of materials to what it should be able to *do* with one, and to specify, in the vocabulary defence planning already uses, what would have to be built for that to be possible under time pressure.

Funding

This research received no external funding.

Data Availability Statement

This is a conceptual paper based on published literature, standards, and policy documents. It generates no new empirical data, and all sources analysed are cited and available publicly. No new datasets were created or analysed.

Disclosure Statement

No potential conflict of interest was reported by the author.

The author read and agreed to the published version of the manuscript.

Use of Artificial Intelligence

During the preparation of this manuscript, the author used DeepL (DeepL SE) to translate the text into English and Claude (Anthropic), current versions between June and September 2026, to language-edit the English text, assist in locating sources and checking factual details, and format the reference list in Harvard style; Claude also helped refine the layout of Figures 1 and 2 (matplotlib diagrams). The author verified all AI-assisted output, including every reference entry against the original source, and takes full responsibility for the content of the manuscript.

References

Black Duck Software (2025) *2025 Open source security and risk analysis (OSSRA) report*. Available at: https://www.blackduck.com/resources/analyst-reports/open-source-security-risk-analysis.html (Accessed: 7 June 2026).

Bush, D.R. (2024) *Software bill of Materials (SBOM)* (policy memorandum). Office of the Assistant Secretary of the Army (Acquisition, Logistics and Technology), Washington, DC, 16 August. Available at: https://api.army.mil/e2/c/downloads/2024/10/17/4072ab1e/asaalt-software-bill-of-materials-policy-signed.pdf (Accessed: 7 June 2026).

Check Point Research (2021) *The numbers behind Log4j vulnerability CVE-2021-44228*. Check Point Software Technologies, Tel Aviv, 13 December (updated 29 December 2021). Available at: https://blog.checkpoint.com/security/the-numbers-behind-a-cyber-pandemic-detailed-dive/ (Accessed: 11 August 2026).

Cybersecurity and Infrastructure Security Agency (CISA) (2021) *Emergency directive 22-02: Mitigate Apache Log4j vulnerability*. CISA, Washington, DC, 17 December. Available at: https://www.cisa.gov/news-events/directives/ed-22-02-mitigate-apache-log4j-vulnerability-closed (Accessed: 7 June 2026).

Cybersecurity and Infrastructure Security Agency (CISA) (2023) *Minimum requirements for vulnerability exploitability eXchange (VEX)*. CISA, Washington, DC. Available at: https://www.cisa.gov/resources-tools/resources/minimum-requirements-vulnerability-exploitability-exchange-vex (Accessed: 11 August 2026).

Cybersecurity and Infrastructure Security Agency (CISA) (2024) *Reported supply chain compromise affecting XZ Utils data compression library, CVE-2024-3094*. CISA, Washington, DC, 29 March. Available at: https://www.cisa.gov/news-events/alerts/2024/03/29/reported-supply-chain-compromise-affecting-xz-utils-data-compression-library-cve-2024-3094 (Accessed: 7 June 2026).

Cybersecurity and Infrastructure Security Agency (CISA) (n.d.) *Software bill of Materials (SBOM)*. CISA, Washington, DC. Available at: https://www.cisa.gov/sbom (Accessed: 7 June 2026).

Enck, W. and Williams, L. (2022) 'Top five challenges in software supply chain security: Observations from 30 industry and government organizations', *IEEE Security & Privacy*, 20(2), pp. 96–100. doi: 10.1109/MSEC.2022.3142338.

European Union (EU) (2022) *Directive (EU) 2022/2555 of the European Parliament and of the Council of 14 December 2022 on measures for a high common level of cybersecurity across the Union (NIS 2 Directive)*,

OJ L 333, 27 December, pp. 80–152. Available at: https://eur-lex.europa.eu/eli/dir/2022/2555/oj/eng (Accessed: 11 August 2026).

European Union (EU) (2024) *Regulation (EU) 2024/2847 of the European Parliament and of the Council of 23 October 2024 on horizontal cybersecurity requirements for products with digital elements (Cyber Resilience Act)*, OJ L, 2024/2847, 20 November. Available at: https://eur-lex.europa.eu/eli/reg/2024/2847/oj/eng (Accessed: 7 June 2026).

European Union Agency for Cybersecurity (ENISA) (2025) *SBOM landscape analysis: Towards an implementation guide, v1.20*. ENISA, Heraklion. Available at: https://www.enisa.europa.eu/sites/default/files/2025-12/SBOM%20Analysis%20-%20Towards%20an%20Implementation%20Guide_v1.20-Published.pdf (Accessed: 30 June 2026).

Farrand, B., Carrapico, H. and Turobov, A. (2024) 'The new geopolitics of EU cybersecurity: Security, economy and sovereignty', *International Affairs*, 100(6), pp. 2379–2397. doi: 10.1093/ia/iiae231.

FireEye (2020) *Highly evasive attacker leverages SolarWinds supply chain to compromise multiple global victims with SUNBURST backdoor*, 13 December. Available at: https://cloud.google.com/blog/topics/threat-intelligence/evasive-attacker-leverages-solarwinds-supply-chain-compromises-with-sunburst-backdoor (Accessed: 7 June 2026).

Fortress Information Security (2024) *SBOM use cases for critical military programs and national security systems*, 16 December. Available at: https://www.fortressinfosec.com/blog/sbom-use-cases-for-critical-military-programs-and-national-security-systems (Accessed: 7 June 2026).

Freund, A. (2024) *Backdoor in upstream xz/liblzma leading to SSH server compromise*. OSS-security mailing list (Openwall.com), 29 March. Available at: https://www.openwall.com/lists/oss-security/2024/03/29/4 (Accessed: 30 June 2026).

Henig, A. and Hyde, C. (2026) *When security scanners become the weapon: Breaking down the Trivy supply chain attack*. Palo Alto Networks, 24 March. Available at: https://www.paloaltonetworks.com/blog/cloud-security/trivy-supply-chain-attack/ (Accessed: 30 June 2026).

International Organization for Standardization (ISO) (2021) *ISO/IEC 5962:2021 information technology: SPDX® specification V2.2.1*. ISO, Geneva. Available at: https://www.iso.org/standard/81870.html (Accessed: 7 June 2026).

Jaakkola, E. (2020) 'Designing conceptual articles: Four approaches', *AMS Review*, 10(1–2), pp. 18–26. doi: 10.1007/s13162-020-00161-0.

Kono, K. and De Tomas Colatin, S. (2023) *National approaches to the supply chain cybersecurity: Taking a more restrictive stance against high-risk vendors*. Tallinn: NATO Cooperative Cyber Defence Centre of Excellence (CCDCOE). Available at: https://ccdcoe.org/uploads/2023/05/Supply_Chain_Cybersecurity.pdf (Accessed: 7 June 2026).

Ladisa, P., Plate, H., Martinez, M. and Barais, O. (2023) 'SoK: Taxonomy of attacks on open-source software supply chains', in *2023 IEEE symposium on security and privacy (SP)*. San Francisco, CA: IEEE, pp. 1509–1526. doi: 10.1109/SP46215.2023.10179304.

Linux Foundation (2022) *The state of Software Bill of Materials (SBOM) and cybersecurity readiness*. San Francisco, CA: The Linux Foundation. Available at: https://www.linuxfoundation.org/research/the-state-of-software-bill-of-materials-sbom-and-cybersecurity-readiness-gated (Accessed: 7 June 2026).

Luenen, C. and Brooks, H. (2024) 'Securing Britain's and NATO's digital supply chains', *NATO Review*, 14 October. Available at: https://www.nato.int/docu/review/articles/2024/10/14/securing-britains-and-natos-digital-supply-chains/index.html (Accessed: 7 June 2026).

Microsoft (2025) *Shai-Hulud 2.0: Guidance for detecting, investigating, and defending against the supply chain attack*. Microsoft Security Blog, 9 December. Available at: https://www.microsoft.com/en-us/security/blog/2025/12/09/shai-hulud-2-0-guidance-for-detecting-investigating-and-defending-against-the-supply-chain-attack/ (Accessed: 30 June 2026).

National Institute of Standards and Technology (NIST) (2022a) *Cybersecurity supply chain risk management practices for systems and organizations.* NIST special publication 800-161, revision 1. Gaithersburg, MD: NIST. doi: 10.6028/NIST.SP.800-161r1.

National Institute of Standards and Technology (NIST) (2022b) *Secure software development framework (SSDF) version 1.1.* NIST special publication 800-218. Gaithersburg, MD: NIST. doi: 10.6028/NIST.SP.800-218.

National Institute of Standards and Technology (NIST) (2024) *The NIST cybersecurity framework (CSF) 2.0.* NIST cybersecurity white paper 29. Gaithersburg, MD: NIST. doi: 10.6028/NIST.CSWP.29.

National Institute of Standards and Technology (NIST) (2025) *CVE-2025-30066: tj-actions/changed-files*. National Institute of Standards and Technology, National Vulnerability Database, Gaithersburg, MD, 15 March. Available at: https://nvd.nist.gov/vuln/detail/cve-2025-30066 (Accessed: 30 June 2026).

National Telecommunications and Information Administration (NTIA) (2021) *The minimum elements for a Software Bill of Materials (SBOM)*. US National Telecommunications and Information Administration, Washington, DC, 12 July. Available at: https://www.ntia.gov/report/2021/minimum-elements-software-bill-materials-sbom (Accessed: 7 June 2026).

North Atlantic Treaty Organization (NATO) (2024) *Defence-critical supply chain security roadmap* (fact-sheet). NATO, Brussels. Available at: https://www.nato.int/content/dam/nato/webready/documents/factsheets/240712-Factsheet-Defence-Supply-Chain-Roadmap-en.pdf (Accessed: 7 June 2026).

North Atlantic Treaty Organization (NATO) (2026) *NATO's role in capability development*. NATO, Brussels, updated 6 July. Available at: https://www.nato.int/en/what-we-do/deterrence-and-defence/natos-role-in-capability-development (Accessed: 11 August 2026).

Ohm, M., Plate, H., Sykosch, A. and Meier, M. (2020) 'Backstabber's knife collection: A review of open source software supply chain attacks', in *Detection of intrusions and malware, and vulnerability assessment (DIMVA 2020)*. Cham: Springer, pp. 23–43. doi: 10.1007/978-3-030-52683-2_2.

Okhravi, H., Burow, N. and Schneider, F.B. (2025) 'Software bill of materials as a proactive defense', *IEEE Security & Privacy*, 23(2), pp. 101–106. doi: 10.1109/MSEC.2025.3528535.

Open Source Security Foundation (OpenSSF) (2023) *Supply-chain levels for software artifacts (SLSA) version 1.0*. OpenSSF, 19 April. Available at: https://slsa.dev/spec/v1.0/ (Accessed: 11 August 2026).

Organization for the Advancement of Structured Information Standards (OASIS) (2022) *Common security advisory framework version 2.0*. OASIS Standard, 18 November. Available at: https://docs.oasis-open.org/csaf/csaf/v2.0/os/csaf-v2.0-os.html (Accessed: 11 August 2026).

OWASP Foundation (2024) *OWASP CycloneDX bill of materials standard (ECMA-424)*. Available at: https://owasp.org/www-project-cyclonedx/ (Accessed: 7 June 2026).

Rid, T. and Buchanan, B. (2015) 'Attributing cyber attacks', *Journal of Strategic Studies*, 38(1–2), pp. 4–37. doi: 10.1080/01402390.2014.977382.

SolarWinds Corporation (2020) *Form 8-K*. US Securities and Exchange Commission, Washington, DC, 17 December. Available at: https://www.sec.gov/Archives/edgar/data/0001739942/000162828020017620/swi-20201217.htm (Accessed: 7 June 2026).

Stalnaker, T., Wintersgill, N., Chaparro, O., Di Penta, M., German, D.M. and Poshyvanyk, D. (2024) 'BOMs away! Inside the minds of stakeholders: A comprehensive study of bills of materials for software systems', in *Proceedings of the IEEE/ACM 46th international conference on software engineering (ICSE '24)*. Lisbon: ACM, pp. 1–13. doi: 10.1145/3597503.3623347.

Tagarev, T. (2009) 'Capabilities-based planning for security sector transformation', *Information & Security: An International Journal*, 24, pp. 27–35. doi: 10.11610/isij.2404.

The White House (2021) *Executive Order 14028: Improving the nation's cybersecurity*, 86 Federal Register 26633, 17 May. Available at: https://www.federalregister.gov/documents/2021/05/17/2021-10460/improving-the-nations-cybersecurity (Accessed: 7 June 2026).

US Cyber Safety Review Board (2022) *Review of the December 2021 Log4j event*. Washington, DC: US Department of Homeland Security. Available at: https://www.cisa.gov/resources-tools/resources/cyber-safety-review-board-csrb (Accessed: 7 June 2026).

US Department of Defense (2024) *Cybersecurity maturity model certification (CMMC) program*, final rule, 32 CFR part 170, Federal Register, 15 October. Available at: https://www.federalregister.gov/documents/2024/10/15/2024-22905/cybersecurity-maturity-model-certification-cmmc-program (Accessed: 11 August 2026).

US Department of Energy (2022) *Cybersecurity capability maturity model (C2M2) version 2.1*. Office of Cybersecurity, Energy Security, and Emergency Response, Washington, DC. Available at: https://www.energy.gov/ceser/cybersecurity-capability-maturity-model-c2m2 (Accessed: 11 August 2026).

US Government Accountability Office (GAO) (2021) *SolarWinds cyberattack demands significant federal and private-sector response* (infographic). US GAO, Washington, DC, 22 April. Available at: https://www.gao.gov/blog/solarwinds-cyberattack-demands-significant-federal-and-private-sector-response-infographic (Accessed: 30 June 2026).

US Joint Chiefs of Staff (2021) *Charter of the joint requirements oversight council (JROC) and implementation of the joint capabilities integration and development system (JCIDS)* (CJCSI 5123.01I). Washington, DC: Joint Chiefs of Staff.

Xia, B., Bi, T., Xing, Z., Lu, Q. and Zhu, L. (2023) 'An empirical study on Software Bill of Materials: Where we stand and the road ahead', in *Proceedings of the 2023 IEEE/ACM 45th international conference on software engineering (ICSE)*. Melbourne: IEEE, pp. 2630–2642. doi: 10.1109/ICSE48619.2023.00219.

Zahan, N., Lin, E., Tamanna, M., Enck, W. and Williams, L. (2023) 'Software bills of materials are required. Are we there yet?', *IEEE Security & Privacy*, 21(2), pp. 82–88. doi: 10.1109/MSEC.2023.3237100.