ACIG
APPLIED CYBERSECURITY & INTERNET GOVERNANCE

NASK

Challenges in DevSecOps Decision-Making amid a Dearth of Valid Frameworks

Francesco Maria Ferazza | Information Security Group, Royal Holloway, University of London, UK | ORCID: 0009-0005-3280-2678

Konstantinos Mersinas | Information Security Group, Royal Holloway, University of London, UK | ORCID: 0000-0002-4402-2987

Abstract

This study examines the challenges of securing DevOps environments through a unique combination of technical framework analysis and behavioural science insights. This is a conceptual analysis based on qualitative review and coding of publicly available DevSecOps frameworks. By analysing frameworks from organisations, such as Open Web Application Security Project, Cloud Security Alliance, the US National Institute of Standards and Technology, and the US Department of Defense, while applying behavioural economics and decision theory, the research investigates how cognitive biases affect security decision-making in DevSecOps and evaluates existing frameworks' gaps. The analysis reveals a significant lack of mature, comprehensive, and regularly updated DevSecOps frameworks, with existing guidelines often lacking clarity, usability, or consideration of human factors. The study identifies key cognitive biases impacting security decisions and demonstrates how these are exacerbated by the absence of robust frameworks. While the research is limited by DevSecOps' evolving nature and ongoing framework development, this limitation itself reflects the field's nascent state and highlights opportunities to observe security

Received: 5.09.2025

Accepted: 27.10.2025

Published: 24.11.2025

Cite this article as:
K. Mersinas, F.M. Ferazza, "Challenges in DevSecOps decision-making amid a dearth of valid frameworks," ACIG, vol. 4, no. 1, 2025, doi: 10.60097/ACIG/213726.

Corresponding author:
Francesco Maria Ferazza, Information Security Group, Royal Holloway, University of London, UK; E-mail: Francesco.Ferazza.2021@live.rhul.ac.uk
0000-0002-4402-2987

practice evolution under uncertainty. Future research could empirically test how framework improvements impact decision-making in real-world DevSecOps environments.

Keywords

security, CI/CD, DevSecops, cognitive biases

1. Introduction

In the landscape of modern software development, delivery, and deployment, the emergence of DevOps methodologies has revolutionised the way organisations approach agility, collaboration, and automation [1–3]. This paradigm shift has been accompanied by a heightened need of – and hence focus on – security, leading to the evolution of DevOps into DevSecOps, an approach that integrates and automates security practices into every phase of the development and delivery life cycle. However, despite the growing recognition of the importance of DevSecOps, security managers are confronted with a multitude of challenges in securing their organisations. Challenges mainly dictated by the novelty of the technologies and processes that enable DevOps. Implementing strong security controls in automated DevOps pipelines requires extensive expertise and informed decisions. Such informed decision-making should be guided by mature and valid frameworks and guidelines. Yet, the current state of DevSecOps frameworks from reputable institutions and bodies reveals a significant lack of maturity and clarity, aside from extremely rare – and limited in scope – cases. Additionally, frameworks and guidelines, including those from influential and reputable organisations, such as the Open Web Application Security Project (OWASP) [4], the Cloud Security Alliance (CSA) [5], the US National Institute of Standards and Technology (NIST) [6], and the US Department of Defense (DoD) [7] often lack consensus and uniformity in their recommendations for securing DevOps workflows. It is worth noting that we purposefully do not analyse vendor-provided guidelines from Cloud Security Providers, for providers such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform often provide guidelines focused around and aimed at selling their own security products. Compounding these challenges are the inherent complexities of decision-making in security management, where cognitive biases can significantly impact the assessment of risks, prioritisation of controls, and the overall security posture. The presence of biases, such as availability bias, anchoring bias, confirmation bias, optimism bias, and the bandwagon effect, can lead security managers astray, resulting in sub-optimal security decisions that may leave organisations vulnerable to both internal and external threats. This

paper seeks to address these issues by conducting a preliminary, yet comprehensive, examination of the human challenges faced by security managers in securing DevOps environments amid a dearth of valid standards and frameworks. Building upon insights from behavioural economics and decision theory, we aim to illustrate the difficulties and the cognitive biases that influence DevOps security decision-making. In order to do so, we imagine a newly appointed security manager in a large enterprise who has to structure and/or revise the security posture of his organisation's DevOps efforts and pipelines. We want to showcase the complexities and the common pitfalls that this hypothetical manager would face, pitfalls greatly exacerbated by the lack of valid, universally accepted, and scrutinised frameworks to be used as reference points and to generate heuristics to simplify the complexity of the task at hand.

The research questions (RQs) we want to shed light on in this paper are the following:

- RQ1: What are the most common cognitive biases and decision-making challenges that security managers face when attempting to secure DevOps environments, and how can these be mitigated or addressed through the use of frameworks and decision support tools?
- RQ2: What is the level of validity, maturity, and completeness of the existing DevSecOps frameworks and guidelines as of early 2024 in terms of providing comprehensive and actionable guidance for security managers to select, implement, and monitor appropriate security controls and practices across the various phases of the DevSecOps lifecycle?
- RQ3: Are there key gaps or limitations in the current DevSecOps frameworks that hinder security managers from making informed decisions regarding their organisation's specific security needs, risks, and emerging technologies or architectures?

While our RQs are conceptual, we identify practical moderators that future empirical work should operationalise: (i) trade-offs among speed, automation, and assurance in the pipeline; (ii) organisational and process maturity; and (iii) contextual risk tolerance. We show where the existing frameworks do – or do not – support practitioners in reasoning about these moderators.

This paper is relevant for three very specific reasons:

1. It serves as a useful snapshot, analysis, and reference point of the current DevSecOps framework landscape for future

research. This is even accentuated by the dramatic need in the job market for DevSecOps trained professionals, with tens of thousands of openings on the main job posting websites for DevSecOps personnel [8].
2. It displays how important a multi-disciplinary approach is in the realm of both information and cybersecurity by stressing how tightly knit the human factor is with all types of technical, logical, and administrative security controls. This is even more true in the DevSecOps context, where culture is a driving force.
3. It serves as a call to action for industry stakeholders to collaborate on the development of robust and universally applicable DevOps security standards, ensuring that the agility and automation of these methodologies are not compromised at the expense of security.

It is worth mentioning that research on the use of frameworks, guidelines, and methodologies to mitigate biases in information security decision-making is not new in literature; however, bibliography explicitly tailored to the context of DevOps is practically non-existant. This is a huge knowledge gap, for the questions being explored are crucial to DevOps, a methodology that has been enabling an unprecedented software development speed, but that is tainted by a huge degree of uncertainty, mainly due to its young age [1, 2], lack of a specific definition and formalisation [5], and overabundance of security products available to decision makers [9]. Additionally, due to its inherently quick software development, delivery, and deployment life cycle, and wide attack surface, DevOps contexts are ontologically more prone to security concerns and risks when compared to other contexts [10–12].

Why decision-making matters?: Framework guidance shapes defaults, trade-offs, and what teams consider 'good enough'. In DevSecOps pipelines, biased judgments (e.g. anchoring on prior controls, optimism and automation bias, and availability from recent incidents) can translate directly into tool sprawl, misallocated spend, and elevated breach exposure. By making these decision points explicit, a framework can reduce error-prone heuristics and improve the consistency and defensibility of security choices.

2. Structure

This paper starts with an introduction to the problem and outlines the research questions. Then it delineates its own structure, terms, and methodology, and illustrates the relevant literature. Then the paper illustrates security decision-making biases,

contextualising them into DevOps environments. The paper then analyses and compares the existing DevSecOps frameworks and guidelines. Afterwards, the paper tries to shed some light on the ability of the examined frameworks to overcome those biases. Finally, conclusions are drawn.

3. Definitions

Before illustrating the methodology applied to the paper, we need to clearly define the terms that will be used.

3.1. Defining the core elements

We need to define the terms 'DevSecOps,' 'framework,' 'bodies and organisations,' and 'usability' of a framework. As far as the term 'DevSecOps' is concerned, we are going to use a reference the NIST definition of it:

> DevSecOps (consisting of acronyms for development, security, and operations, respectively) is one of the facilitating paradigms for the development, deployment, and operation of applications with primitives such as continuous integration, continuous delivery, and continuous deployment (CI/CD) pipelines. [13]

Meanwhile, whenever we refer to 'Framework', we mean a structured set of guidelines, best practices, and tools that help organisations manage and improve their DevSecOps posture. For 'bodies and organisations', we refer to entities and institutions, both public and private which play a specialised role in helping other actors ensuring their DevOps security. As far as what constitutes a 'valid' framework, since the scope of this paper is not to propose a brand new and finely tuned way to assess a framework's validity, we define it by analysing the basic salient dimensions that a framework should have. These dimensions and their weight are explained in a following section.

4. Methodology

In this section, we explain the methodology applied in this paper. To answer RQ1, we identify biases and place them in the DevSecOps context. To answer RQ2, we establish criteria for choosing specific bodies and organisations to obtain frameworks and guidelines from, and we establish how the validity of frameworks and guidelines are assessed. To answer RQ3, we will take a

'complementary' approach, explaining how a perfectly valid and ideal framework would help overcome most biases and then proceed to explain that such a framework does not exist in the real world, ultimately showing what available frameworks lack.

Study type and scope: This is a conceptual paper. We review publicly available documents from major organisations and use deductive thematic coding to identify gaps and recurring patterns in DevSecOps security decision-making and cognitive biases. Our aim is to describe framework maturity and the decision support they offer; we do not present new empirical data.

4.1. Identify and contextualise biases

To tackle RQ1, we outline the main difficulties and cognitive biases in cybersecurity decision-making and frame them in the context of DevOps/DevSecOps. The paper draws on the vast literature in decision-making, cognitive psychology, and behavioural economics to identify challenges decision makers face, providing real-world scenarios and examples of each bias. Specifically, we reviewed key cognitive biases documented in decision-making literature to determine which biases are most pertinent to DevSecOps security decisions. We prioritised biases widely recognised to skew risk judgements under uncertainty (e.g. anchoring and confirmation bias) [14, 15] and those observed in cybersecurity contexts (e.g. optimism bias in security planning [16]). Through collaborative deliberation, the authors reached a consensus to focus on five principal biases – availability bias, anchoring bias, confirmation bias, optimism bias, and the bandwagon effect – for in-depth analysis and contextualisation within the DevSecOps scenario, while omitting other biases that were considered less directly applicable.

4.2. Criteria for choosing specific bodies and organisations

There are many organisations and bodies that provide security guidance in the IT world. In order to answer RQ2, we defined a specific subset of these to analyse their DevSecOps frameworks. We only account for widely scrutinised and reputable bodies; hence, we select only bodies and organisations that meet the following criteria:

- Industry recognition: The institution is recognised by industry experts, peers, and other organisations, indicating a strong reputation in the field. Recognition can come in many forms, such as awards, certifications, or partnerships with other organisations.

- Transparency: The institution is transparent about its operations, policies, and procedures. Institutions that publicly publish their research, methodologies, and findings are more likely to be trustworthy.
- Expertise: The institution has a team of experts with relevant experience and qualifications. Such institutions with demonstrable expertise are more likely to provide reliable and accurate information.
- Collaboration: The institution collaborates with other organisations, researchers, and experts. Such collaboration suggests a broader perspective and often more comprehensive information.
- DevSecOps guidelines: The institution has published (or is in the process of publishing) publicly available DevSecOps frameworks or guidelines.

Bodies and organisations can be public or private. (As a note, we do not analyse guidelines and frameworks created or sponsored by Cloud Service Providers or product vendors, to keep the scope product agnostic).

4.3. Assessing the frameworks

As mentioned above, the purpose of this paper is not to create a finely tuned methodology to assess a framework's validity from scratch. However, since we do need to assess framework validity, we focus on a set of specific criteria or dimensions to obtain a basic qualitative understanding of a framework's viability, soundness, and defensibility. This methodology is applied to answer RQs 2 and 3. We deliberately use a minimal set of criteria as lenses, rather than a full metric: they are sufficient to compare maturity and decision support without implying false precision. The criteria are the following:

- Comprehensiveness, scope, and depth: This dimension assesses the breadth and depth of the framework's coverage. It involves evaluating the range of security domains (e.g. network security, application security, and data security) addressed by the framework and the completeness of controls within each domain.
- Age: This involves assessing the framework's development history, frequency of updates or revisions, and its adaptation to emerging threats and technologies. Older frameworks that have undergone more iterations and refinements potentially indicate a higher level of maturity.
- Community support: This considers the level of community engagement, collaboration, and support surrounding the

framework. It includes the availability of community-contributed resources, forums, and training, which can indicate the framework's adoption and the collective experience of its users.
- Continuous improvement and adaptability: This reflects the framework's ability to evolve in response to changing threats, regulations, and business requirements. Mature frameworks typically have mechanisms for ongoing review, feedback, and enhancement to ensure they remain relevant and effective over time.

We do not assign numeric scores: given heterogeneous scope and update cadence, lightweight qualitative comparison avoids pseudo quantification and improves reproducibility. These criteria provide an encompassing yet pragmatic approach for qualitatively evaluating a framework's viability. They align with attributes identified in prior work as critical for effective security frameworks: broad coverage of relevant security domains (comprehensiveness) [17], evidence of iterative refinement and updates over time (maturity) [18], active engagement and support by a community of practitioners (community support) [19], and the capacity to adapt to emerging threats and requirements (adaptability) [18]. The authors agreed on focusing on these four dimensions through collaborative discussions, aiming to capture both technical depth and real-world applicability.

4.4. Linking frameworks and biases

Data collection: In this study, we examined publicly available DevOps security frameworks, focusing on both technical and human-centric measures. Our analysis involved three steps: data collection, qualitative coding, and thematic analysis.

Qualitative coding: We then performed a qualitative, deductive coding process, annotating relevant text segments for content indicating direct or indirect coverage of cognitive biases. For example, if a framework described processes for prioritising new or high-profile threats, we flagged those references to see whether they might align with mitigating or reinforcing the *availability bias*. Similarly, if a framework suggested regularly updating initial threat assessments, we coded that as a measure that potentially counteracts the *anchoring bias*. By iterating through these documents and refining the codes, we built an initial map of how each framework addressed various bias-related concerns.

Thematic analysis: After coding, we grouped similar codes under themes reflecting specific biases, for example, availability,

anchoring, confirmation, optimism, and bandwagon. By comparing segments across frameworks, we identified patterns and gaps. For example:

- Availability bias*:* References to 'recently publicised exploits' being prioritised over less visible vulnerabilities were placed under this theme, revealing whether frameworks encouraged balanced risk evaluations.
- Anchoring bias*:* Any prescribed process for re-evaluating initial threat assessments or shifting security strategies over time was coded here, indicating proactive steps to avoid overreliance on first impressions.

This deductive thematic analysis allowed us to identify how frameworks might either reinforce or mitigate these cognitive biases. We then linked each identified gap to potential strategies for bias mitigation, drawing from both existing literature and DevSecOps best practices.

These insights informed the subsequent discussion of what an 'ideal' DevSecOps security framework could encompass in order to address the full spectrum of human and organisational factors.

5. Literature review

DevSecOps has rapidly gained attention as an approach to embed security into DevOps workflows, but there is still no universally accepted framework or standard for its implementation. Multiple industry and government bodies have proposed guidelines or models – for example, the OWASP DevSecOps guideline [4], the CSA's 'Six pillars of DevSecOps' [5], as well as initiatives by NIST [6], and the US DoD [7] – yet these efforts differ in scope and focus, reflecting a lack of consensus on best practices.

In academia, researchers have attempted to consolidate DevSecOps knowledge through systematic reviews. One such review identified a range of challenges (spanning people, process, tools, and infrastructure) that practitioners face when adopting DevSecOps and emphasised the need for better developer-centric security tools and a balance between rapid delivery and security [20]. Similarly, a recent multivocal literature review distilled five primary dimensions of DevSecOps (definitions, challenges, practices, tools, and metrics) and proposed a unified model integrating these aspects [21], while also noting gaps such as the underexplored 'global' context of DevSecOps adoption. Despite these efforts, the literature indicates

that DevSecOps guidance remains fragmented and immature, with organisations often left to navigate disparate frameworks and ad hoc practices.

To ensure comprehensive coverage of relevant literature, methodologies such as citation chaining and cross-verification were employed. Citation chaining, both forward and backward, leverages reference lists of key studies and identifies subsequent research citing those studies, helping to capture literature that might be missed by keyword searches alone [22]. Cross-verification, involving comparing results from multiple databases (primarily IEEE Xplore, ACM Digital Library, and Scopus), helps ensure completeness and mitigate selection bias [23].

Cognitive and decision-making biases have been extensively studied in psychology and are known to skew rational judgment under uncertainty [14, 15, 24]. Biases such as the confirmation bias, anchoring, availability bias, and overconfidence can lead individuals to misjudge risks or overlook critical information in cybersecurity contexts. A growing body of work documents how such biases impede effective security decision-making. For instance, optimistic bias (an illusion of invulnerability) has been observed among security professionals [16], and researchers have noted that common heuristics and biases can negatively influence the design and selection of security controls [25]. Some studies have explored ways to mitigate these effects: for example, leveraging behavioural economics principles to improve security decisions [26] or employing decision support systems to counteract bias in risk assessment [27]. Notably, biases are pervasive even in general software engineering practice – one field study found about 70% of developer actions were associated with at least one cognitive bias, with developers resorting to ad hoc means to cope due to lack of systematic support [28]. These findings underscore that human biases pose a serious challenge in any complex high-stakes decision environment.

Given these two streams of knowledge, it is striking that few studies have examined how cognitive biases intersect with DevSecOps. Integrating security into a fast-moving DevOps pipeline is as much a human challenge as a technical one, yet most DevSecOps research focuses on tools and processes, rather than human factors [29]. It stands to reason that the DevSecOps context could amplify certain decision biases: the pressure for speed and continuous delivery might encourage reliance on mental shortcuts, and heavy automation could engender overreliance on tools (automation bias). Likewise, a strong 'DevOps culture' might unintentionally

foster groupthink or confirmation bias if teams become too aligned with prevailing assumptions. Furthermore, the abundance of new technologies and often contradictory advice in this domain creates cognitive overload, making security professionals more susceptible to biases when selecting controls or prioritising threats [30–32]. However, to date, there is virtually no published research explicitly exploring how cognitive biases play out in DevSecOps practices.

Although cognitive biases are a well-documented driver of suboptimal cybersecurity decision-making, broader cultural and organisational factors also play a important role. Established cross-cultural frameworks – notably Hofstede's cultural dimensions theory [33] and Meyer's *The culture map* [34] – demonstrate how, amongst other variables, variations in power distance, uncertainty avoidance, communication style, and risk tolerance influence perceptions and behaviours which can be expanded in security contexts [35]. However, by design, this paper remains focused exclusively on psychological biases; a deeper examination of cultural influences lies beyond its scope.

In summary, while robust DevSecOps frameworks and guidelines are still evolving, and cognitive biases are known to influence security decisions, their intersection remains largely uncharted. Current DevSecOps guidance rarely accounts for the human element of decision-making, and we lack empirical understanding of a how biases might hinder (or be mitigated by) DevSecOps tools and culture. This gap in the literature suggests that further investigation is needed to determine how cognitive biases affect security outcomes in DevSecOps environments, and how future frameworks might integrate awareness of these biases to better support security professionals.

Building on the foundational literature of cognitive psychology and decision-making, this paper subsequently focuses on several key biases – namely availability bias, anchoring bias, confirmation bias, optimism bias, and the bandwagon effect – that are particularly pertinent to the fast-paced and complex nature of DevSecOps. The following analysis section then specifically contextualises these selected biases, demonstrating their impact within typical DevSecOps decision-making scenarios.

6. Analysis

6.1. On cognitive biases and decision-making

In this first analysis section we explore the challenges and the cognitive pitfalls that a security manager might fall for when

securing DevOps processes. This also serves the purpose of underlining why the human aspect is a central one in cyber and information security, especially in this context. DevOps being a rather new and constantly evolving methodology, it is to be expected that the average security manager might not be fully aware of the best practices needed to achieve the desired security posture. This is further exacerbated by the colossal amount of tools available on the market to provide DevOps security controls; as of 2024, the Cloud Native Computing Foundation can count over 900 products [9, 36]. In a technology procurement context characterised by significant information asymmetry, security managers are often overwhelmed by the multitude of available solutions, making it difficult to navigate and choose effectively. This asymmetry in procurement occurs because vendors typically have more detailed knowledge about the capabilities, limitations, and potential vulnerabilities of their products than buyers do. This disparity can lead to challenges for security managers who must rely on the information provided by vendors, which may be biased or incomplete, in such a scenario making well-informed decisions becomes difficult, increasing the risk of selecting inadequate security solutions. The abundance of complex and often contradictory information further exacerbates the problem, creating an environment where security managers struggle to identify the most suitable and effective tools for their specific needs [30–32]. It is widely accepted and self-evident that when there is a lack of knowledge and expertise, security frameworks and guidelines are usually the quickest and safest reference for decision makers to get a holistic, normalised, and comprehensive understanding of the security goals, controls, and mechanisms relevant to their environments. However, as we've illustrated in the previous sections, while there's a breadth of pertinent frameworks, most are not fully usable, maintained, publicly scrutinised, and mature. All critical qualities aid decision makers in a scientifically sound way.

6.1.1. Decision-making challanges

Securing DevOps/DevSecOps contexts presents significant challenges for decision-makers. As mentioned earlier, one difficulty is navigating the vast landscape of security tools and technologies available, each designed to address specific risks and vulnerabilities at various stages of the software development life cycle (SDLC) and deployment pipeline. Additionally, Identifying the optimal set of security controls and integrating them seamlessly into the intricate workflows of the DevOps pipeline can be a daunting task. For instance, decision-makers must evaluate the appropriate placement of security gates, such as static application security testing (SAST), dynamic

application security testing (DAST), software composition analysis (SCA), and infrastructure-as-code (IaC) security scanners, to ensure comprehensive coverage without impeding the velocity and agility of the DevOps processes. The decision on where to place controls is closely related to fully understanding how a DevOps pipeline is structured and what the inputs, outputs, and risks of each of its phases are. This can be a challenge itself, for DevOps pipelines can vary widely across organisations, incorporating different tools, technologies, and processes at each stage, from code development and integration, from building to testing, deploying, and monitoring phases. This complexity makes it difficult to clearly delineate security responsibilities and accountabilities across teams and stakeholders, further exacerbating the challenges in implementing effective DevSecOps practices. Furthermore, the complexity of modern architectures and deployment environments, including containerised applications, server less functions, and cloud-native micro services, presents unique security challenges. Decision-makers must have a deep understanding of these technologies and their associated security implications to implement effective security measures, such as container image scanning, runtime security monitoring, and cloud security posture management (CSPM) tools. Compounding these challenges is the difficulty in defining and measuring relevant security metrics and key risk indicators (KRIs) that accurately reflect the security posture of the DevOps pipeline. Without a clear understanding of what metrics to track and how to interpret them, decision-makers may struggle to assess the effectiveness of their security controls and make informed decisions about areas that require improvement or additional investment. Moreover, the automation and continuous integration/continuous deployment (CI/CD nature of DevOps environments requires security controls to be tightly integrated into the automated processes and pipelines. This necessitates a deep understanding of automation frameworks, scripting languages, and Application Programming Interfaces (APIs) to ensure seamless integration and real-time monitoring of security events and incidents. Lastly, the very essence of DevOps, with its emphasis on collaboration and shared responsibilities across development, operations, and security teams, can pose challenges in terms of aligning priorities, establishing clear lines of accountability, preventing turf wars, and fostering effective communication and coordination among these diverse stakeholders, each with their own technical backgrounds and perspectives.

6.1.2. Decision-making biases

As highlighted in the literature review section, cognitive biases present significant challenges in decision-making. This section now provides a detailed contextualisation of five prominent

biases previously identified – availability bias, anchoring bias, confirmation bias, optimism bias, and the bandwagon effect – within the DevSecOps domain. We explore how our hypothetical and newly appointed security manager would encounter these pitfalls when trying to make sense of the complicated DevSecOps landscape. Each bias is supported by a short description drawn from established research, followed by an example specific to DevSecOps, and a brief explanation of how a sound and solid framework could help mitigate these cognitive challenges.

6.1.3. Availability bias

Description: This bias occurs when individuals overestimate the importance of information that is readily available to them [37–39]. In the context of DevSecOps decision-making, security managers may prioritise security incidents or risks that have received recent attention, even if they are not the most significant threats to their organisation.

Impact and example: Due to the lack of mature and comprehensive DevSecOps frameworks, security managers may rely heavily on recent security incidents or anecdotes when making decisions about which security controls to implement in their DevOps pipelines. For instance, in the wake of high-profile supply chain attacks like the solar winds breach [40, 41], security managers might allocate disproportionate resources to implementing SCA tools and hardening their software supply chains, even if their organisation's primary risks lie elsewhere, such as in misconfigured cloud infrastructure or insufficient monitoring and logging practices. Rather than being reactive, a more proactive and risk-based approach would involve conducting thorough threat modeling and risk assessments to identify the most relevant DevSecOps technologies and controls, such as IaC security scanners, container security tools, or centralised secrets management solutions.

6.1.4. Anchoring bias

Description: The anchoring bias (or effect) occurs when individuals are heavily influenced by the first piece of information encountered when making decisions, although it might not be relevant or important [15, 42, 43]. In the context of DevSecOps, this bias might cause security managers to fixate on initial risk assessments or security control recommendations, even if new information suggests a reassessment is necessary.

Impact and example: Despite the emergence of new container-based architectures and the associated security risks, security

managers might persistently rely on their initial decisions to implement traditional security controls, such as network firewalls and web application firewalls (WAFs), overlooking the need for specialised container security solutions like image scanning, runtime monitoring, and admission control policies. Similarly, initial risk assessments that prioritised securing the development and testing phases might anchor security managers' decisions, causing them to overlook the growing importance of securing the IaC and continuous deployment processes, leaving the organisation vulnerable to misconfigurations and insecure deployments.

6.1.5. Confirmation bias

Description: Confirmation bias involves seeking out or interpreting information in a way that confirms preexisting beliefs or hypotheses [44–46]. In DevSecOps decision-making, this bias might lead security managers to selectively interpret evidence that supports their assumptions about the effectiveness of certain security controls.

Impact and example: A security manager might strongly believe that a traditional WAF is sufficient to secure their application infrastructure. Despite warnings from security experts about the limitations of WAFs, the manager continues to rely heavily on the WAF and dismisses evidence that highlights the need for comprehensive security measures tailored to containers. This biased decision-making can leave significant security gaps, as WAFs do not address the unique challenges of securing containers, such as vulnerabilities in container images or misconfigurations in container orchestration. Ignoring comprehensive container security measures weakens the organisation's security posture, making it vulnerable to container-specific attacks, such as those exploiting runtime vulnerabilities.

6.1.6. Optimism bias

Description: Optimism bias leads individuals to underestimate the likelihood of negative events occurring [47–49]. In the context of DevSecOps, security managers may underestimate the probability of security breaches or vulnerabilities affecting their DevOps pipelines.

Impact and example: In DevSecOps, managers may rely too much on their defence in depth stack with tools like Jenkins, Docker, and Aqua Security to protect CI/CD pipelines. This optimism can cause problems: insufficient resources for security checks, misprioritised controls, and neglect of continuous monitoring, leading to delayed

breach detection. In 2020, many companies delayed patching an high CVSS Jenkins vulnerability, mostly due to their optimistic views on their CI/CD defence in depth, ultimately resulting in breaches [50, 51].

6.1.7. Bandwagon effect

Description: The bandwagon effect, also known as groupthink, occurs when individuals align their beliefs or behaviours with those of a larger group, a bias fostered by peer benchmarking approaches that large companies often employ [52–54]. In DevSecOps decision-making, this bias might cause security managers to adopt popular security practices or solutions without critically evaluating their suitability for their organisation's specific context. A typical example of this is the so-called news-centric approach to information security [55].

Impact and example: Security managers may overlook alternative approaches or innovative solutions that could better address the unique security challenges of their DevOps environment, opting instead to follow prevailing trends or industry norms. For example, in response to peer pressure or industry hype, security managers may consider SAST tools as their primary line of defence against codebase-related vulnerabilites, despite such tools being known for their extremely high false positive rates [56].

In the following section we analyse the frameworks from the chosen bodies, and, afterwards, we briefly compare them.

6.2. Framework analysis

Applying the five criteria mentioned in the methodology section, we now analyse the DevSecOps frameworks and guidelines from these bodies: OWASP, CSA, the US NIST, and the US DoD.

Other reputable bodies were considered, but failed to meet one or more of the required criteria. For example, the European Union Agency for Cybersecurity (ENISA) recognises DevOps as a critical emerging methodology in its 'European Cybersecurity Skills Framework,' but does not provide a guideline or framework to secure such methodology. It is also worth reiterating that some cloud service providers do offer DevSecOps frameworks, but our intention is to keep the paper product-agnostic and only based on information that is free from commercial interests.

In the following sections, we briefly analyse each single framework by looking at the chosen dimensions and characteristics and then compare the four frameworks.

6.2.1. OWASP DevSecOps framework

The OWASP DevSecOps guideline explains 'how we can implement a secure pipeline and use best practices and introduce tools that we can use in this matter. Also, the project tries to help promote the shift-left security culture in our development process' [4]. This framework mainly splits the DevSecOps effort into seven domains, namely:

1. Init: Dedicated to the initial phases of the creation of a DevSecOps pipeline, with a strong focus on the human aspect of DevSecOps, this phase deals with training and security champions.
2. Pre-commit: Dedicated to everything that happens before committing code, such as linting, thread modelling, and secrets management.
3. Commit/continuous integration: Focused on the phase of code integration and SAST controls.
4. Continuous delivery: Dedicated to the phase of continuous delivery, with controls, such as DAST, misconfiguration checks, and API security.
5. Continuous deployment: Dedicated to code deployment, keys, and certificates management.
6. Operations: Focused on monitoring production environment, with pentesting, logging, bug bounties, and attack simulations.
7. Governance: This last domain is centred on compliance auditing, data protection, and reporting.

This framework was created in 2020 and was last updated in March 2023 via GitHub, and, like most OWASP projects, it is fully open and relies on public contribution for its updates. Despite the framework being 4 years old, and accepting external contributions, only a few domains appear to contain any information at all, namely the second and third domain have meaningful guidelines and checklists, whereas everything else is at an empty stub status; hence, this framework cannot be deemed complete.

6.2.2. NIST DevSecOps framework

While the actual content of this framework is not yet available, we decided to analyse the logic behind it, as well as its design and scope, because NIST is, arguably, one of the most reputable

bodies when it comes down to security frameworks. The US NIST launched this framework's project in 2021, and in 2022 published the relevant project description, a major work that will likely require a few more years to be completed [6]. The key features of this framework are the following:

1. The framework revolves around a risk-based approach.
2. The framework covers two completely different scenarios, a free and open source software (FOSS) one, and a closed source one. This matches the needs of SMEs and larger enteprises, for the former may opt for FOSS due to its cost-effectiveness and flexibility, while the latter may prefer closed-source software for its comprehensive support, advanced features, regulatory compliance, and vendor accountability.
3. It will try to map security requirements and controls to those outlined in other NIST frameworks, such as the 'framework for improving critical infrastructure cybersecurity', the 'risk management framework' (RMF), the 'secure software development framework' (SSDF), and the 'workforce framework for cybersecurity' (NICE framework).

We have no way to determine when the final product will be released, how often the framework will be updated, and how. Previous NIST security frameworks are not updated by a community-led effort, rather from a top-down initiative, and this framework will most likely receive a similar treatment. While the domains of this framework are still unknown, its project description paper allows us to understand that it will cover, at least the following areas:

1. Human resources management, for clearly defined roles, responsibilities, and skill sets represent a critical security condition.
2. Security by design and security by default.
3. Both static and dynamic code base testing.
4. Endpoint and IDE security.
5. Strong focus on compliance, auditability, accountability, and logging, as per normative and regulatory requirements.

It is worth noting that this framework is heavily US-centric, with risks, security controls, and security mechanisms defined by other NIST special publications, executive orders, and the US federal laws. This might devalue the framework for non-US-based organisations and entities looking for country-agnostic guidelines.

6.2.3. CSA DevSecOps framework

Cloud Security Alliance efforts to provide a DevSecOps framework started back in 2019 with the publication of the overview of their framework; 'with DevSecOps still in its infancy, there are still questions surrounding how it should be structured. CSA is working to provide best practices and guidance to help organisations effectively implement DevSecOps' [5, 57]. The CSA DevSecOps framework is based on six domains, or pillars, as the CSA named them:

1. Collective responsibility: This pillar focuses on ensuring that every person within an organisation feels responsible about security, not seeing it as someone else's job. Everyone is a security champion, and security is not separate from business objectives.
2. Collaboration and integration: Focuses on the cultural aspects of security. This pillar aims to create a security-aware and collaborative (non-confrontational) culture within the organisation. This pillar stresses how humans are the weakest line of the security chain.
3. Pragmatic implementation: Focuses on giving guidance on security tools procurement and implementation, trying to help security managers understand what qualities to look for in the solutions they want to integrate in their organisation.
4. Bridging compliance and development: Regulatory and compliance teams care more about having a process in place than checking every step of it. On the other hand, DevOps teams think the code itself proves everything, so they don't focus as much on documenting processes. The fourth pillar aims to bridge this gap.
5. Automation: This pillar focuses on automating security practices, reducing the testing and feedback loops, and eliminating as many non-automatable tasks as possible.
6. Measure, monitor, report, and action: The last pillar focuses on a staple of DevOps, measuring as many actionable metrics as possible, and report and act on those as quickly as possible to achieve a safer posture.

This framework references ISO 27000 series to define most of its security objectives and controls, making it easily understandable by most professionals and not tied to the regulatory and normative requirements of a specific country or industry sector. The second pillar was released in February 2024, while the sixth pillar of this framework is not yet available and there is no expected release

date. Most of this framework relies on a CSA working group for its updates. There are open discussion groups on the CSA website and open online meetings that are used to shape the document.

6.2.4. DoD DevSecOps guidelines

The US DoD released an unclassified version for public use of these guidelines in early 2021 [7]. This document is designed with DoD DevSecOps teams and DoD DevOps capabilities providers in mind. These guidelines revolve around defining each phase of the DevOps life cycle, and maps each phase, using a taxonomy, to a set of supporting tools, security objectives, and security activities. The guidelines split DevSecOps efforts into 10 phases: general security, planning, developing, building, testing, deliver, deploy, operations, monitoring, and configuration management. For each of those phases the document shows a wide array of tools, the features those tools should offer, and the benefits of using such tools. It is worth noting that this document is completely product-agnostic, tools are described by their functionality, and neither commodity nor commercial tool is ever explicitly, nor implicitly, named. Most of the definitions for the security objectives, activities, and controls present in this guideline are taken from the SSDF NIST special publication [58]. This guideline has not been update since 2021, at least in its publicly available version; it is developed internally by the US DoD, without any public scrutiny or discussion.

7. Results

7.1. Frameworks comparison and analysis

The most glaring aspect emerging from the analysis of the frameworks is that only the DoD document is complete, and provides a complete taxonomy of each and every DevOps phase and their associated security controls, objectives, and activities. All other frameworks are either incomplete works in progress (OWASP and CSA), or, as far the NIST guideline is concerned, nothing has been published yet. Among these unfinished ones, only the CSA guidelines provide a clear estimate for the release of the finished product; OWASP and NIST do not commit to any specific deadline. Another noteworthy difference between these frameworks is that the NIST and DoD ones, being created within the US federal government, are designed around the US federal laws, regulations, and NIST definitions. This could be a limit for security managers looking for country-agnostic frameworks. Let us now consider a very useful dimension, the frequency by which we can expect these frameworks to be updated. This is an important dimension to consider, because

DevOps is in a constant flow of evolution, and so are threats and malicious actors that may harm DevOps contexts; as such, frameworks need to adapt as quickly as possible to this rapidly changing technical and adversarial landscape. Frameworks and guidelines open to public contribution, such as the OWASP and the CSA ones are more likely to be updated more frequently; this is mainly due to their less rigid structure and lack of a large bureaucratic overhead. The OWASP framework especially seems like a good candidate for routine and frequent updates due to OWASP having an active and engaged community, and previous OWASP deliverables with similar functions have been – and still are – updated often [59, 60]. The last pivotal difference between frameworks lies in how many different domains and phases they partition the DevOps security efforts into, and how in depth they go within each domain. While the structures of the domains are rather heterogeneous between frameworks, the topics covered are fairly similar for the most part. All analysed documents stress the importance of good project scoping, security by design, pre-commit (e.g. integrated development environments security tools, Git hooks), and post-commit (e.g. static and dynamic code analysis) controls, supply chain controls (e.g. SCA and container security), integration tests, operational security, monitoring, and measuring security-relevant metrics. However, there are differences in how these are treated and explained. The NIST framework still has to be released, so we only know the areas of interest it will be focused on, without any notion of their intended depth and structure. OWASP gives very thorough and step-by-step guidelines on how to secure each of its phases are; however, not every phase is available yet. The OWASP framework has a strongly technical focus, some of its phases are also illustrated, and this kind of visualisation can be useful [4]. The CSA document, while still not finished, is well balanced, covering both technical aspects of security controls in its 'Pragmatic Implementation' pillar, but also managerial, compliance, and legal aspects. The US DoD guideline is the most solid and well structured among the considered documents. It thoroughly illustrates all the tools, activities, and security objectives of the phases it conceives. This guidelines document, however, falls short of a very important – if not central – of DevSecOps, the human aspect. This one aspect is completely ignored by the DoD document, while it is clearly stated as important and treated by the other three examined documents. The CSA framework, dedicates several of its six pillars to the human aspect of DevSecOps, analysing how central a cultural shift in security is in DevSecOps, and how collaboration and understanding between different entities within the same organisation, for example, developers and compliance (pillar 4) is pivotal to mission success, to enhance security.

It is worth noting that the human and cultural aspect is a central one in DevOps and DevSecOps. In fact, most practitioners use the acronym CALMS [61] to describe these new methodologies. The C – the very first letter – stands for 'culture', for a cultural shift in how security is managed and incidents are treated (and expected) is a critical and enabling factor of these new approaches to software development and delivery. As such, it seems that the DoD document is missing a crucial part. Finally, the NIST and DoD frameworks and guidelines are heavily US-centric, referencing NIST special publications and executive orders and aimed at compliance with the US laws and regulations. This might make them ill suited for use for activities outside the United States. The key point of this section is that, while all the analysed frameworks represent valid initiatives and share many common and useful pillars, they either lack some critical aspects of securing DevOps, such as with the DoD document ignoring human aspects, or they are unfinished, or they are not routinely updated, or publicly scrutinised. All these characteristics would make a complete, defensible, and future-proof framework. Table 1 shows a quick recap of the salient dimensions we chose to analyse the examined frameworks.

8. Discussion

In this section, we first examine the implications of our findings and highlight the critical shortcomings in the existing frameworks and the consequences of neglected human and organisational factors. We then introduce the notion of an ideal framework that could comprehensively cover all dimensions of security, followed by detailed strategies for mitigating specific cognitive biases. Finally, we acknowledge the limitations of our research and suggest directions for future investigation, including empirical validation and a more quantitative assessment of DevSecOps frameworks.

8.1. Implications of findings

Our findings underscore critical shortcomings in the existing frameworks. Without comprehensive, regularly updated, and vetted guidance, organisations remain vulnerable to biases such as, but not limited to, availability, anchoring, confirmation, and overconfidence biases. The incomplete nature of these frameworks could lead to inconsistent security practices across organisations, significantly increasing the likelihood of security incidents and the magnitude of their impact. Furthermore, the evident neglect of human and organisational factors within several frameworks

Table 1. Framework differences.

Organisation	Complete	Public contribution	Age (years)	Latest update	Domains and depth	References
OWASP	No	Yes	5	2023	7/Detailed	OWASP/MITRE
CSA	No	Yes	5	2024	6/Detailed	ISO
NIST	No	Limited	0	None	N/A	US regulations and NIST SP
DOD	Yes	No	3	None	10/extremely detailed	US regulations and NIST SP

indicates a systemic oversight that could exacerbate security vulnerabilities and compromise organisational resilience. Addressing these gaps is critical, not only to enhance security practices, but also to cultivate a robust security culture within organisations.

8.2. Ideal framework and bias mitigation

Our previous analysis revealed limitations in the existing frameworks: None were fully finalised, routinely updated, and publicly vetted. Furthermore, coverage varied between frameworks. For example, the DoD guidelines completely ignore the human aspect of the DevOps paradigm in toto.

Hypothetical ideal framework: This section assumes the existence of a complete and valid framework encompassing all desirable characteristics mentioned above, including covering all aspects of security, ranging from technical controls to human aspects. We can view this ideal framework's coverage as the sum of the coverage of all analysed frameworks, and we explore how such a framework could serve as a valuable decision-making tool able to mitigate – to some extent – the aforementioned biases. Below is a detailed explanation of how such a framework could assist in mitigating such biases.

8.2.1. Availability bias mitigation strategy

Comprehensive threat intelligence integration: Continuously aggregating and analysing threat intelligence from diverse sources ensures that decision makers have access to a wide array of information beyond recent high-profile incidents. This helps to provide a balanced view of potential risks.

Automated risk prioritisation: Utilising machine learning algorithms, the framework allows for automatically prioritising risks based on their potential impact and likelihood, rather than their

recent prominence. This ensures that resources are allocated to the most significant threats, rather than the most visible ones.

Example implementation: The framework would necessitate the implementation of a security product featuring advanced dashboard functionality. This dashboard would dynamically rank threats by analysing historical data, current threat landscapes, and potential organisational impact. Such a system would guide security managers to prioritise the most critical issues, rather than being swayed by recent events and, thus, mitigating the availability bias.

8.2.2. Anchoring bias mitigation strategy

1. Dynamic risk-assessment tools: The framework should feature tools that continuously reassess risks as new information becomes available. This ensures that initial assessments are regularly updated and do not disproportionately influence ongoing decisions.
2. Decision review mechanisms: Implementing regular review sessions where initial decisions are re-evaluated in light of new data can help mitigate the influence of early information.

Example implementation: The framework prompts periodic reassessments of security controls and risks, incorporating the latest data on threats and vulnerabilities, and provides alerts when significant changes occur that warrant a review of the existing security measures.

8.2.3. Confirmation bias mitigation strategy

1. Diverse data sources and analytical tools: By incorporating a wide range of data sources and using analytical tools that challenge the existing assumptions, the framework helps ensure that decisions are based on a comprehensive view of the evidence.
2. Peer review and collaboration: Encouraging a culture of peer review and collaboration within the framework can expose decision makers to alternative viewpoints and counteract the tendency to seek out confirming information.

Example implementation: The framework includes features for collaborative threat modelling and security control assessment, where teams can provide feedback and challenge assumptions, ensuring a balanced and unbiased evaluation of security measures. Collaborative analysis, peer reviewing, and security participation are a prominent staple of DevOps, so they are more relevant in this context than in non-DevOps ones, as such an ideal framework would most likely structure and leverage these activities.

8.2.4. Optimism bias-mitigation strategy

1. Scenario planning and simulation: The framework should incorporate tools for running realistic security breach simulations and scenario planning, helping managers to better understand the likelihood and impact of negative events.
2. Regular vulnerability assessments: Implementing mandatory, regular vulnerability assessments and penetration tests to provide an ongoing reality check against overly optimistic security assessments.

Example implementation: The framework schedules and conducts regular simulated attacks and vulnerability assessments, providing reports that highlight potential weaknesses and necessary improvements, thus counteracting undue optimism. Attack simulations and assessments, while being a common cyber security practice, are even more relevant in DevOps contexts due to their inherently larger attack surface, mostly dependant from microservices-heavy infrastructures.

8.2.5. Bandwagon effect-mitigation strategy

1. Customisable security solutions: The framework offers customisable security solutions tailored to the specific context of the organisation, rather than promoting one-size-fits-all approaches.
2. Critical evaluation tools: Providing tools for critical evaluation of security practices and trends, including cost-benefit analyses and risk assessments tailored to the organisation's unique environment.

Example implementation: The framework includes a module that assesses the effectiveness of popular security practices in the context of the organisation's specific environment, helping managers to make informed decisions based on their unique needs, rather than industry trends. This is extremely relevant for DevOps practices because architectures, infrastructures, technical resources, and software development methodologies and strategies can greatly vary between organisation, more than in non-DevOps contexts. It is noteworthy that an action or control suggested by a framework might be able to counter several of these biases at once, for instance, recommending the use of structured analytic techniques to interpret both CTI feeds, vulnerability assessments, and penetration tests would greatly impact most cognitive biases related to understanding complex data and gauging risks [62]. It is also worth remembering that, as stated in the introductory section, some of these mitigations and controls also apply to generic,

non-DevOps-bound security contexts, but they are more relevant within the DevOps domain [5, 9–12].

8.2.6 More ideal framework control examples

Table 2 presents additional concrete examples of human- and engineering-level controls – beyond those outlined in the preceding bias-specific mitigation strategies – that an ideal DevSecOps framework could adopt to mitigate anchoring, confirmation, optimism, availability, and bandwagon biases. Rather than exhaustively cataloguing every possible control, it illustrates how targeted interventions can systematically reduce cognitive biases and strengthen decision-making and security posture in practice.

8.3. Research limitations

This is a conceptual paper based on qualitative reading of publicly available framework materials, not new empirical data, so some judgements reflect interpretation. To compare heterogeneous and fast-evolving frameworks in a fair way, we deliberately kept a small set of qualitative criteria and avoided numeric scores to prevent false precision. The traits we propose for an 'ideal' framework are therefore hypotheses, not yet validated in the wild.

8.4. Future research

Future research should empirically validate the conceptual claims of this paper by combining qualitative case studies (e.g. interviews with DevSecOps practitioners) with quantitative assessments (e.g. a maturity scoring model) to benchmark framework effectiveness. Beyond cognitive biases and core human factors, such as collaboration and communication, it is important that subsequent studies explicitly account for cross-cultural variability. Established cultural frameworks – most notably Hofstede's cultural dimensions theory [33] and Meyer's *The culture map* [34] – demonstrate how national and organisational culture shapes risk tolerance, decision-making styles, and information-sharing. Integrating these cultural dimensions into the design and evaluation of DevSecOps frameworks improves their contextual relevance and applicability across diverse environments.

9. Conclusion

Our analysis shows that while existing DevOps security frameworks often address various technical controls, they remain incomplete and lack ongoing public validation, particularly

Table 2. DevSecOps controls targeting specific cognitive biases.

Control	Example	Mitigated bias/how it helps
Human dimension		
Decision reappraisal Sessions	Structured 'fresh-eyes' reviews of initial requirements	Anchoring bias: breaks initial anchors by forcing reassessment
Cross-functional peer reviews	Developers paired with security specialists for code review	Confirmation bias: surfaces contradictory evidence through diverse perspectives
Pre-mortems	Team identifies failure scenarios before work begins	Optimism bias: highlights risks early to counter overconfidence
Devil's advocate Role	Rotate a designated dissent champion in planning meetings	Bandwagon effect: encourages dissent and reduces conformity pressure
Periodic Assumption Checkpoints	Scheduled milestone reviews of core assumptions	Anchoring bias: revisits and validates initial decisions over time
Anonymous Feedback Channels	Secure surveys for reporting concerns anonymously	Bandwagon effect: allows dissent without peer pressure
Security incident Debriefs	Structured post-incident sessions including low-severity events	Availability bias: counters recency by reflecting on all incidents
Engineering dimension		
Pre-commit controls	Git pre-commit hooks enforce linting, secret scanning, formatting, and static analysis before commits are accepted	Anchoring bias: catches unexamined assumptions and skipped tests early
Automated SAST/DAST Scans	CI pipeline runs independent static/dynamic scans on every pull request	Confirmation bias: uncovers issues missed by developer expectations
Chaos engineering Tests	Inject controlled failures into staging environments	Optimism bias: reveals hidden system fragility before production
Policy-as-code enforcement	Enforce security policies automatically via IaC tools	Bandwagon effect: prevents 'everyone does it this way' by codifying best practice
Independent security audits	Third-party automated code scanning compliance checks and	Confirmation bias: introduces impartial scrutiny to challenge internal assumptions
Risk contingency Buffers	Automated alerting for unpatched vulnerabilities with scheduled remediation windows	Optimism bias: counters overly optimistic assumptions by enforcing time-bound fixes
Automated dependency vulnerability management	Automated dependency scanning with prioritised patch recommendations in CI/CD	Bandwagon effect: avoids herd mentality by surfacing data-driven patch priorities

concerning the human and organisational factors vital to effective security practices. Many are ephemeral or not fully finalised, limiting their broader adoption and real-world applicability. By highlighting key cognitive biases – availability, anchoring, confirmation, optimism, and the bandwagon effect – and mapping them to mitigation strategies, we demonstrate that technical solutions alone cannot address these biases; instead, structured processes, clear guidance, and continuous re-evaluation of decisions are also needed.

We outline the components of an 'ideal framework' that integrates technical controls with human-centric considerations to close these gaps. Although further empirical work is required to measure the real-world effectiveness of bias-focused interventions, this study underscores the need for frameworks that evolve over time and incorporate feedback from diverse security environments.

Addressing our RQs, this paper identified critical gaps in current DevSecOps frameworks (RQ1), highlighted cognitive biases as significant barriers to effective security practices (RQ2), and outlined essential characteristics of an ideal integrated framework that addresses both technical and human-centric challenges (RQ3). Additionally, by clearly delineating these issues and proposing specific strategies, we set the foundation for future research aimed at refining and empirically validating integrated DevSecOps frameworks. As a conceptual synthesis, our analysis motivates small-N case studies and field experiments to test how specific framework affordances reduce biased decisions and improve security outcomes in practice.

References

[1] G. Kim, K. Behr, G. Spafford, *The phoenix project: A novel about it, DevOps, and helping your business win*. Portland, OR: IT Revolution Press, 2013.

[2] G. Kim, *The unicorn project: A novel about developers, digital disruption, and thriving in the age of data*, 1st ed. Portland, OR: IT Revolution Press, 2019.

[3] M. Rajkumar, A.K. Pole, V.S. Adige, P. Mahanta, "DevOps culture and its impact on cloud delivery and software development." in *International Conference on Advances in Computing, Communication, & Automation (ICACCA)* (Spring), Dehradun, India, Apr 2016. New York, NY: IEEE, 2016, pp. 1–6, doi: 10.1109/ICACCA.2016.7578902.

[4] Open Web Application Security Project (OWASP), *DevSecOps guideline*, original date: 2020-05-15T15:45:51Z. Wilmington, DE: OWASP Foundation, 2024.

[5] Cloud Security Alliance (CSA), *Six pillars of DevSecOps*. Bellingham, WA: CSA, 2024.

[6] Computer Security Resource Center (CSRC), I.T.L. Computer Security Division, *DevSecOps.* Gaithersburg, MD: NIST, CSRC, Oct. 2020.

[7] US Department of Defense (DoD), *DoD DevSecOps fundamentals and activities*. Arlington, VA: US Department of Defense, 2024.

[8] *LinkedIn (2024). DevOps Jobs in United States.* [Online]. Available: https://www.linkedin.com/jobs/search/?keywords=devops\&location=United\%20States [Accessed: Mar. 20, 2025].

[9] Cloud Native Computing Foundation (2024). *CNCF Landscape*. [Online]. Available: https://landscape.cncf.io/ [Accessed: Mar. 20, 2025].

[10] A. Rahman, C. Parnin, L. Williams, "The seven sins: Security smells in infrastructure as code scripts," in *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, Montreal, QC, Canada, May 2019. New York, NY: IEEE, 2019, pp. 164–175, doi: 10.1109/ICSE.2019.00033.

[11] A. Rahman, R. Mahdavi-Hezaveh, L. Williams, "A systematic mapping study of infrastructure as code research," *Information and Software Technology*, vol. 108, pp. 65–77, 2019, doi: 10.1016/j.infsof.2018.12.004.

[12] N. Mateus-Coelho, M. Cruz-Cunha, L.G. Ferreira, "Security in microservices architectures," *Procedia Computer Science*, vol. 181, pp. 1225–1236, 2021, doi: 10.1016/j.procs.2021.01.320.

[13] R. Chandramouli, *Implementation of DevSecOps for a microservices-based application with service mesh*, Tech. Rep. NIST SP 800-204C. Gaithersburg, MD: National Institute of Standards and Technology, 2022, doi: 10.6028/NIST.SP.800-204C.

[14] D. Kahneman, *Thinking, fast and slow*, 1st ed. New York, NY: Farrar, Straus and Giroux, 2011.

[15] A. Tversky, D. Kahneman, "Judgment under uncertainty: Heuristics and biases: Biases in judgments reveal some heuristics of thinking under uncertainty," *Science*, vol. 185, no. 4157, pp. 1124–1131, 1974, doi: 10.1126/science.185.4157.1124.

[16] H.-S. Rhee, Y. Ryu, C.-T. Kim, "I am fine but you are not: Optimistic bias and illusion of control on information security," in Proceedings of the International Conference on Information Systems, Association for Information Systems, Atlanta, GA, ICIS 2005, Dec. 11–14, 2005, Las Vegas, NV. 2005.

[17] S. Naskar (2024). *Why NIST in cybersecurity*? GRC Docs blog. [Online]. Available: https://grc-docs.com/blogs/nist-faq/why-nist-in-cybersecurity. [Accessed: Mar. 20, 2025].

[18] National Institute of Standards and Technology (NIST). *Framework for improving critical infrastructure cybersecurity, version 1.1*, Cybersecurity White Paper. Gaithersburg, MD: NIST, 2018.

[19] Wiz Blog (Academy). (2023). *Application security frameworks and standards: OWASP, NIST, ISO/IEC*. [Online]. Available: https://www.wiz.io/academy/application-security-frameworks. [Accessed: Mar. 20, 2025].

[20] R.N. Rajapakse, M. Zahedi, M.A. Babar, H. Shen, "Challenges and solutions when adopting DevSecOps: A systematic review," *Information and Software Technology*, vol. 141, p. 106700, 2022, doi: 10.1016/j.infsof.2021.106700.

[21] X. Zhao, T. Clear, R. Lal, "Identifying the primary dimensions of DevSecOps: A multi-vocal literature review," *Journal of Systems and Software*, vol. 214, no. C, p. 112063, 2024, doi: 10.1016/j.jss.2024.112063.

[22] J. Hirt, T. Nordhausen, C. Appenzeller-Herzog, H. Ewald, "Citation tracking for systematic literature searching: A scoping review," *Research Synthesis Methods*, vol. 14, no. 1, pp. 26–38, 2023, doi: 10.1002/jrsm.1635.

[23] T. Horsley, O. Dingwall, M. Sampson, "Checking reference lists to find additional studies for systematic reviews," *The Cochrane Database of Systematic Reviews*, vol. 2011, no. 8, p. MR000026, 2011, doi: 10.1002/14651858.MR000026.pub2.

[24] B. Fischhoff, P. Slovic, S. Lichtenstein, "Knowing with certainty: The appropriateness of extreme confidence," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 3, no. 4, pp. 552–564, 1977, doi: 10.1037/0096-1523.3.4.552.

[25] V. Garg, J. Camp, "Heuristics and biases: Implications for security design," *IEEE Technology and Society Magazine*, vol. 32, no. 1, pp. 73–79, 2013, doi: 10.1109/MTS.2013.2241294.

[26] S.L. Pfleeger, D.D. Caputo, "Leveraging behavioral science to mitigate cyber security risk," *Computers & Security*, vol. 31, no. 4, pp. 597–611, 2012, doi: 10.1016/j.cose.2011.12.010.

[27] G. Phillips-Wren, D.J. Power, M. Mora, "Cognitive bias, decision styles, and risk attitudes in decision-making and DSS," *Journal of Decision Systems*, vol. 28, no. 2, pp. 63–66, 2019, doi: 10.1080/12460125.2019.1646509.

[28] S. Chattopadhyay, N. Nelson, A. Au, N. Morales, C. Sanchez et al., "Cognitive biases in software development," *Communications of the ACM*, vol. 65, no. 4, pp. 115–122, 2022, doi: 10.1145/3517217.

[29] M.-L. Sánchez-Gordón, R. Colomo-Palacios, "Security as culture: A systematic literature review of DevSecOps," in *Proceedings of 1st International Workshop on Engineering and Cybersecurity of Critical Systems*, ICSE 2020 Workshop, ICSEW'20: Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops, pp. 266-269, Seoul, South Korea. 2020. doi: 10.1145/3387940.3392233.

[30] R. Anderson, "Why information security is hard – An economic perspective," in *Seventeenth Annual Computer Security Applications Conference*, New Orleans, LA. New York, NY: IEEE Computer Society, 2001, pp. 358–365, doi: 10.1109/ACSAC.2001.991552.

[31] R. Anderson, T. Moore, "The economics of information security," *Science*, vol. 314, no. 5799, pp. 610–613, 2006, doi: 10.1126/science.1130992.

[32] M. Brecht, T. Nowey, "A closer look at information security costs," in *The economics of information security and privacy*, R. Böhme, Ed. Berlin: Springer, 2013, pp. 3–24, doi: 10.1007/978-3-642-39498-0_1.

[33] G. Hofstede, *Culture's consequences: International differences in work-related values*. Beverly Hills, CA: Sage, 1980.

[34] E. Meyer, *The Culture map: Breaking through the invisible boundaries of global business*. New York, NY: Public Affairs, 2014.

[35] M. de Bruin, K. Mersinas, "Individual and contextual variables of cyber security behaviour – An empirical analysis of national culture, industry, organisation, and individual variables of (in) secure human behaviour," *arXiv preprint, arXiv:2405.16215*, 2024, doi: 10.48550/arXiv.2405.16215.

[36] Cloud Native Computing Foundation (2024). *CNCF Landscape*. [Online]. Available: https://landscape.cncf.io/. [Accessed: Mar. 20, 2025].

[37] D. Ariely, *Predictably irrational: The hidden forces that shape our decisions*, revised and expanded ed. 3. [print]. New York, NY: Harper Collins, 2009.

[38] A. Tversky, D. Kahneman, "Availability: A heuristic for judging frequency and probability," *Cognitive Psychology*, vol. 5, no. 2, pp. 207–232, 1973, doi: 10.1016/0010-0285(73)90033-9.

[39] H.A. Simon, "A behavioral model of rational choice," *The Quarterly Journal of Economics*, vol. 69, no. 1, p. 99, 1955, doi: 10.2307/1884852.

[40] Reuters. (Feb. 2021). *Solar winds hack was "largest and most sophisticated attack" ever: Microsoft president*, US. Available: https://www.reuters.com/article/idUSKBN2AF03Q/. [Accessed: Mar. 20, 2025].

[41] US GAO Office. (Mar. 2024). *Solar winds cyberattack demands significant federal and private sector response (infographic)*. [Online]. Available: https://www.gao.gov/blog/solarwinds-cyberattack-demands-significant-federal-and-private-sector-response-infographic. [Accessed: Mar. 20, 2025].

[42] D. Ariely, I. Simonson, "Buying, bidding, playing, or competing? Value assessment and decision dynamics in online auctions," *Journal of Consumer Psychology*, vol. 13, nos. 1–2, pp. 113–123, 2003, doi: 10.1207/S15327663JCP13-1&2_10.

[43] C. Camerer, G. Loewenstein, D. Prelec, "Neuroeconomics: How neuroscience can inform economics," *Journal of Economic Literature*, vol. 43, no. 1, pp. 9–64, 2005, doi: 10.1257/0022051053737843.

[44] R.S. Nickerson, "Confirmation bias: A ubiquitous phenomenon in many guises," *Review of General Psychology*, vol. 2, no. 2, pp. 175–220, 1998, doi: 10.1037/1089-2680.2.2.175.

[45] K.E. Stanovich, R.F. West, "Individual differences in reasoning: Implications for the rationality debate?" *Behavioral and Brain Sciences*, vol. 23, no. 5, pp. 645–665, 2000, doi: 10.1017/S0140525X00003435.

[46] H. Mercier, D. Sperber, *The enigma of reason*. Cambridge, MA: Harvard University Press, 2017.

[47] N.D. Weinstein, "Unrealistic optimism about future life events," *Journal of Personality and Social Psychology*, vol. 39, no. 5, pp. 806–820, 1980, doi: 10.1037/0022-3514.39.5.806.

[48] T. Sharot, *The optimism bias: A tour of the irrationally positive brain*, 1st ed. New York, NY: Pantheon Books, 2011, OCLC: ocn667609433.

[49] D.A. Armor, S.E. Taylor, "Situated optimism: Specific outcome expectancies and self-regulation," *Advances in Experimental Social Psychology*, vol. 30, pp. 309–379, 1998, doi: 10.1016/S0065-2601(08)60386-X.

[50] *Openwall. Multiple vulnerabilities in Jenkins plugins.* [Online]. Available: https://www.openwall.com/lists/oss-security/2020/09/23/1 [Accessed: Mar 20, 2025].

[51] *National Institute of Standards and Technologies (NIST). CVE-2020-2279.* [Online]. Available online: NVD - CVE-2020-2279. [Accessed: Mar. 20, 2025].

[52] T. Kuran, *Private truths, public lies*. Cambridge, MA: Harvard University Press, 1998, doi: 10.2307/j.ctvt1sgqt.

[53] C.R. Sunstein, "The law of group polarization," *Journal of Political Philosophy*, vol. 10, no. 2, pp. 175–195, 2002, doi: 10.1111/1467-9760.00148.

[54] L. Festinger, "Informal social communication," *Psychological Review*, vol. 57, no. 5, pp. 271–282, 1950, doi: 10.1037/h0056932.

[55] R. Brown, S.J. Roberts, *Intelligence-driven incident response: Outwitting the adversary*, 2nd ed. Beijing: O'Reilly, 2023, OCLC: on1390610883.

[56] B. Aloraini, M. Nagappan, D.M. German, S. Hayashi, Y. Higo, "An empirical study of security warnings from static application security testing tools," *Journal of Systems and Software*, vol. 158, p. 110427, 2019, doi: 10.1016/j.jss.2019.110427.

[57] Cloud Security Alliance (CSA), *Six pillars of DevSecOps series.* Bellingham, WA: CSA, 2024.

[58] Computer Security Resource Center (CSRC), I.T.L. Computer Security Division. *Secure software development framework.* Gaithersburg, MD: NIST, CSRC, 2021.

[59] Open Web Application Security Project (OWASP), *WSTG-latest.* Wilmington, DE: OWASP Foundation, 2024.

[60] Open Web Application Security Project (OWASP), *OWASP risk assessment framework.* Wilmington, DE: OWASP Foundation, 2024.

[61] Atlassian, *CALMS framework*. *San Francisco, CA:* Atlassian, 2024.

[62] R.J. Heuer, R.H. Pherson, *Structured analytic techniques for intelligence analysis.* Washington, DC: CQ Press, 2011.