Automatic Cross-chain Protocol Pausing with Real-time Conformance Checking

Bruno Mateus
INESC-ID, Instituto Superior Técnico
Universidade de Lisboa
Lisboa, Portugal
bruno.mateus@tecnico.ulisboa.pt

André Augusto
INESC-ID, Instituto Superior Técnico
Universidade de Lisboa
Lisboa, Portugal
andre.augusto@tecnico.ulisboa.pt

Rafael Belchior
Blockdaemon / INESC-ID
Lisboa, Portugal
rafael.belchior@tecnico.ulisboa.pt

André Vasconcelos
INESC-ID, Instituto Superior Técnico
Universidade de Lisboa
Lisboa, Portugal
andre.vasconcelos@tecnico.ulisboa.pt

Miguel Correia
INESC-ID, Instituto Superior Técnico
Universidade de Lisboa
Lisboa, Portugal
miguel.p.correia@tecnico.ulisboa.pt

***Abstract*—The proliferation of diverse blockchain networks has intensified the demand for interoperability, but cross-chain bridges introduce significant security risks. This paper proposes a protocol protection framework for early incident detection and containment through a runtime transactions conformance checking pipeline. Deviations from expected behavior trigger a pausing mechanism, halting operations to prevent incorrect state propagation across blockchains. Our evaluation shows that anomalies can be detected and cross-chain asset transfers paused in less than 10 seconds, reducing attack exposure and preventing cascading failures. The results highlight how automated pausing is a critical safeguard for securing interoperability infrastructures and strengthening cross-chain ecosystems.**

***Index Terms*—Blockchain Interoperability, Cross-Chain Security, Incident Detection, Incident Response**

I. Introduction

Few technological breakthroughs have attracted as much attention as blockchain technology since its introduction in Nakamoto's seminal whitepaper [1]. However, the rapid growth of heterogeneous platforms has created a fragmented ecosystem [2], forcing users to choose between networks and limiting the scalability of decentralized applications. To overcome this, cross-chain bridges have emerged as a key interoperability mechanism, but their vulnerabilities have made them prime targets for exploits [3], [4]. High-profile incidents such as the Ronin and Nomad attacks caused losses of $611M and $190M, respectively [5], underscoring the urgency for robust security mechanisms. Unlike centralized systems that follow standardized response frameworks such as the NIST "Computer Security Incident Handling Guide" [6], cross-chain attacks require coordination across multiple networks, creating complexities beyond conventional approaches [7].

This research addresses the lack of automated safeguards for incident containment in cross-chain protocols [5]. We propose a real-time conformance checking system that monitors transaction flows during execution and halts bridge operations upon the detection of anomalies, enabling early incident detection and containment, reducing the risk of cascading failures. The main contributions are: (i) the design and implementation of real-time conformance checking for cross-chain protocols and (ii) the development of an automated protocol pausing mechanism to prevent the propagation of malicious activity across blockchain networks.

Hyperledger Cacti[1] is an open-source interoperability framework from the Linux Foundation[2] that provides a modular, plugin-based architecture for cross-network operations, enabling integration of heterogeneous ledgers. Among these is the Secure Asset Transfer Protocol (SATP) [8], [9], which ensures atomic and secure asset transfers between chains via specialized bridge applications called *gateways* that include recovery models for handling failures [10]. Building on this foundation, Hephaestus [11] is a Cacti plugin that generates models (*ccmodels*) from cross-chain transfers to verify expected behavior. While originally limited to post-facto analysis, our work extends Hephaestus to perform runtime conformance checking, enabling immediate detection of protocol deviations during execution.

II. Cross-chain Security and Incident Response

A. CrossChain Risk Framework

Although cross-chain communication protocols work to enable seamless data exchange and asset transfer, they also introduce risks that have been categorized by the *Cross-chain*

This work was financially supported by the Project Blockchain.PT – Decentralize Portugal with Blockchain Agenda, (Project no 51), ref. C632734434-00467077, WP 7: Interoperability, Call no 02/C05-i01.01/2022, funded by the Portuguese Recovery and Resilience Program (PRR), The Portuguese Republic and The European Union (EU) under the framework of Next Generation EU Program. This work was also supported by national funds, through Fundação para a Ciência e a Tecnologia I.P. (FCT) under projects UID/50021/2025 and UID/PRR/50021/2025.

[1]https://github.com/hyperledger/cacti/
[2]https://www.hyperledger.org/

Risk Framework [12] into *Network Consensus*, *Protocol Architecture*, *Protocol Operational*, and *Protocol Implementation* risks. These risks reflect the challenges of relying on the state validity and finality of the underlying chains, protocol design trade-offs, complex integration of on- and off-chain components in protocol implementation, and secure management of privileged credentials, respectively. Without proper safeguards, such as monitoring, anomaly detection and emergency pausing, incidents can cascade across interconnected networks.

B. Bridge Vulnerabilities and Defense Strategies

Cross-chain bridge vulnerabilities span multiple layers of complexity, encompassing technical flaws, architectural weaknesses, and operational oversights. Common vulnerabilities include *private key leakage*, which can grant unauthorized access to privileged operations; *front-end phishing*, where attackers manipulate the interface to mislead users; and *authentication bypass*, enabling illegitimate users to interact with the bridge. Issues such as *incorrect proof verification* and *malicious event emission* allow attackers to submit invalid data or simulate legitimate transactions. Additional threats arise from *uncontrolled access to sensitive functions*, *flawed business logic*, and *dependencies on unverified external contracts*, all of which can lead to significant financial and operational damage. Even initial deployment phases are not immune, as *inaccurate initialization logic*, such as misconfigured admin roles, can expose the bridge to persistent threats. To address these challenges, mitigation strategies such as *multi-signature key management*, *interface security enhancements*, *strong access control mechanisms*, *formal verification*, and *secure deployment practices* are critical. Cross-chain bridge security can be improved by adopting best practices, including robust access controls, transaction thresholds, pause mechanisms, consistent error handling with rollback protocols, and regular security audits with bug bounty programs to proactively identify and mitigate vulnerabilities while maintaining system integrity [3], [4], [13]–[15].

C. Incident Response and Handling

Incident handling in traditional settings is supported by Computer Security Incident Response Teams (CSIRTs) [16], [17], often following the phases of *Detection* (identifying potential threats), *Triage* (prioritizing and categorizing incidents), *Analysis* (assessing impact and root causes), and *Response* (containment, eradication, and recovery) [6], [18]. In blockchain interoperability, containment is especially critical since immutable operations cannot be undone.

Our work focuses on a practical mechanism with detection and containment of incidents through the ability to *pause* protocols, temporarily halting sensitive operations such as asset minting or burning, to limit the spread of malicious activity until remediation is possible.

III. Automated Asset Transfer Pausing

Cross-chain protocols are essential for blockchain interoperability, but their interconnected nature means that incidents can spread rapidly across systems. Existing solutions often lack automated detection or depend on manual oversight, leading to delayed responses. To address this, we extended Hephaestus with runtime conformance checking and integrated it with SATP, enabling automatic pausing of bridge operations when anomalies are detected. Although designed for the Hyperledger Cacti ecosystem, the approach is applicable to other protocols. The source code is available on GitHub[3].

A. Cross-Chain Model Conformance Checking

Hephaestus was upgraded to monitor transactions in real-time across Hyperledger ledgers connectors. Using Reactive Extensions for JavaScript (RxJS) for asynchronous message passing and Process Mining for Python (PM4PY) to generate cross-chain models that represent legitimate execution flows, incoming events are continuously compared against them and deviations are flagged at runtime (Fig. 1). This enables immediate detection of nonconforming behavior instead of retrospective analysis.

For example, if a model describes the correct transfer process as `lockAsset`, `burnAsset`, and `mintAsset`, and a user mints an asset directly after locking it, the initial `lockAsset` will align with the model, but when the `mintAsset` event is detected without a prior `burnAsset`, it is flagged as nonconformant behavior, marking the transaction as incorrect.

[3]https://github.com/brunoffmateus/blockchain-integration-framework/tree/hephaestus-satp-integration

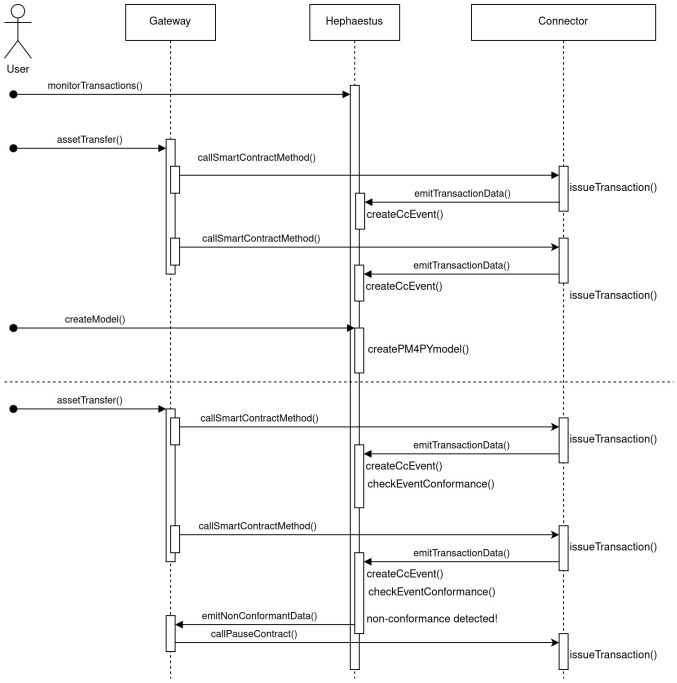


Fig. 1. Asset Transfer Overview

B. Automated Pausing of SATP Contracts

To contain detected violations, we added admin-controlled `pause` functions to the SATP bridge contracts deployed in the ledgers. When a nonconformance event is detected, the system notifies all subscribed gateways through a RxJS subscription-based system (Fig. 2), but only those with admin permissions are authorized to pause bridge contracts, preventing unintended actions. This ensures that malicious or faulty behavior is stopped before propagating to other networks, prioritizing consistency and containment over availability.

IV. Evaluation

Our evaluation had two goals: (1) assess Hephaestus' ability to generate cross-chain models and (2) validate real-time misbehavior detection and automated pausing. The evaluation was carried out on a Lenovo IdeaPad running Ubuntu 20.04.6 LTS, equipped with 13.5 GB of RAM and an AMD Ryzen 7 5700U CPU. All tests were implemented using the Jest framework, with local instances of Hyperledger Besu, Ethereum, and Fabric, connected through Cacti.

A. Cross-Chain Model Pipeline Evaluation

To understand the impact of workload on model creation in Hephaestus, we devised three use cases involving 1, 10, and 50 asset transfers from Fabric to Besu, repeating each test 15 times. As shown in Fig. 3, the transaction modeling phase, in which gateways emit transactions to the blockchains and Hephaestus captures events, scaled with workload size, accounting for 75% of the total time in the largest test case. The increased latency can be attributed to ledger interactions and the messaging cycle between the SATP gateways, not Hephaestus itself, since Hephaestus handled transactional data asynchronously, confirming that the primary scalability bottleneck lies in transaction issuing.

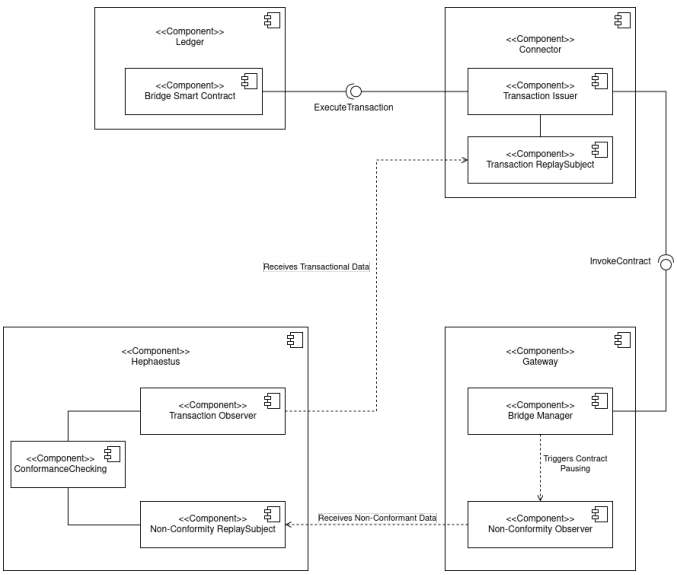


Fig. 2. Component View of the Incident Detection Mechanism

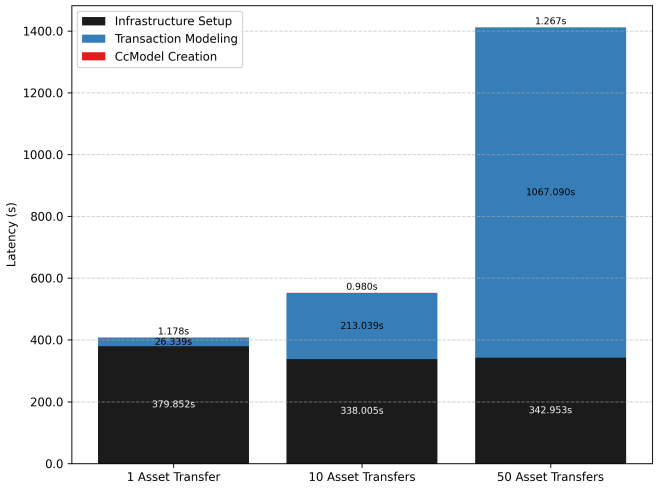


Fig. 3. Cross-Chain Model Pipeline Performance

Meanwhile, the synchronous operation of generating the cross-chain model remains extremely lightweight across all scenarios, with its latency never exceeding 0.3% of total runtime, indicating that it is independent of workload size. This demonstrates the suitability for runtime model generation and its efficiency in high-throughput environments. Nevertheless, Hephaestus, like any system aiming to derive a reliable ground truth, requires at least one complete asset transfer, and additional transfers only improve the model's accuracy, if the protocol supports multiple valid execution paths.

B. SATP Gateway Pausing Evaluation

We focused on adversaries who attempt to exploit protocol violations for financial gain or disruption. Each violation has distinct impacts: `skip-lock` and `skip-burn` cause duplication and fund loss, `double-mint` creates inflation, and `skip-mint` and `skip-assign` result in asset inaccessibility or loss of ownership. To evaluate resilience against these attacker strategies, each violation scenario was implemented and tested 15 times across different ledgers.

Transaction conformance checks averaged ∼1s (Fig. 4), confirming that conformance checking introduces negligible overhead to execution, which suggests that Hephaestus handles deviations in protocol behavior with minimal impact on performance. Contract pausing averaged 7.6s (Fig. 5), with maximum pausing times consistently remaining below 10 seconds, demonstrating reliable responsiveness under varying ledger infrastructure.

Hephaestus consistently detected 100% of the violations, automatically pausing bridge contracts to contain the attack. This shows that Hephaestus effectively mitigates workflow logic attacks by continuously comparing observed executions against administrator-defined models and enforcing containment upon deviations. However, attacks that do not manifest as execution-flow violations, such as misuse of administrative keys or malicious event emission that still conforms to the model, remain out of scope, underscoring that Hephaestus

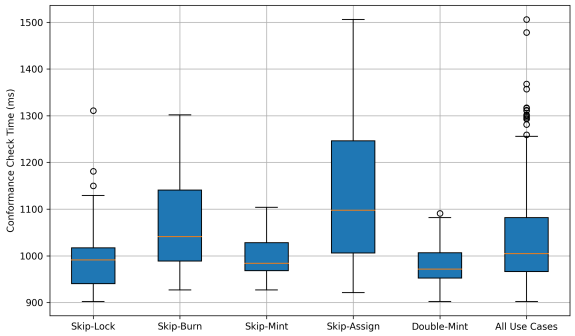


Fig. 4. Latency Boxplots for the Conformance Checks of each Use Case

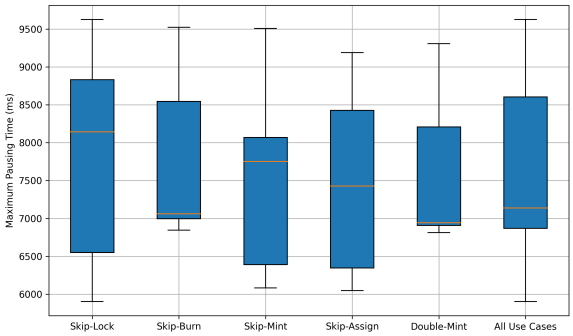


Fig. 5. Latency Boxplots for the Automated Pausing of each Use Case

complements rather than replaces other security mechanisms such as key management, protocol validation, and audits.

C. Detection Capabilities and Limitations

Our approach can help mitigate some bridge vulnerabilities described in Section II-B, as it acts as a behavioral enforcement layer that checks whether the observed behavior is consistent with the correct execution. *Incorrect Proof Verification* and *Privilege Escalation* can be mitigated by enforcing observed execution models at runtime and flagging events that disrupt logical flows. *Authentication Bypass* and *Uncontrolled Access*, are not directly mitigated but may still be detected through deviations in execution flow; however, these issues lie not with Hephaestus, but with poorly designed protocol validation and access mechanisms.

However, Hephaestus' conformance checking is not a comprehensive defense mechanism. It remains susceptible to *Malicious Event Emission*, since it does not verify the source of chain transactions and may therefore produce false negatives. It is also vulnerable to *Incorrect Business Logic*, as flawed but consistent execution paths become part of the learned models. These limitations highlight that runtime monitoring is best viewed as a complementary safeguard, and robust protocol design, formal audits, and input validation remain essential for achieving comprehensive security.

As with any automated system, our approach has limitations. Detection also occurs only after a transaction is confirmed to the blockchain, meaning some damage may take place; this could be addressed by integrating Hephaestus with rollback-capable protocols which could enable automated recovery. There is a trade-off between containment and availability, as pausing contracts can delay user transfers or complicate rollbacks. Future work could delegate rollback logic to dedicated contracts or explore a pre-submission validation model, though the latter would introduce new trust assumptions and weaken blockchain immutability guarantees.

Hephaestus enforces only the set of events specified in the model, defined by administrators according to the use case. In practice, this means that only events deemed critical (e.g., mint, burn) need to be included in the model, while benign administrative operations can simply be excluded and not trigger pausing, inherently reducing the risk of false positives. However, we acknowledge that a systematic evaluation including benign and adversarial operations would allow the computation of precision, recall, and confusion matrices, which we consider an important direction for future work.

Overall, this research demonstrate that runtime conformance checking combined with automated pausing provides rapid and reliable containment of protocol violations, enhancing the resilience of cross-chain interoperability.

V. Conclusion

This paper addresses a key challenge in cross-chain interoperability: securing protocols against unexpected or malicious behavior. By integrating runtime conformance checking with reactive transaction pausing, we propose an effective defense against faults and attacks that could propagate across networks. We enhanced Hephaestus with real-time conformance checking and integrated it with SATP to enable automated bridge pausing, containing violations in under 10 seconds. These contributions not only strengthen Hyperledger Cacti but also provide a generalizable design for other interoperability frameworks.

Future work could focus on refining conformance checking (e.g., context-aware checking to avoid false positives), enabling parallel monitoring for high-throughput environments, supporting multiple cross-chain models could allow validation of different types of operations, and integrating rollback mechanisms could enable automatic recovery. Our work showed that combining real-time conformance checking with automated pausing can greatly enhance protocol safety and reliability, helping advance the goal of secure and seamless interoperability across heterogeneous blockchains.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *http://www.bitcoin.org/bitcoin.pdf*, 2009.

[2] S. Schulte, M. Sigwart, P. Frauenthaler, and M. Borkowski, "Towards blockchain interoperability," in *Business Process Management: Blockchain and Central and Eastern Europe Forum: BPM 2019 Blockchain and CEE Forum, Vienna, Austria, September 1–6, 2019, Proceedings 17*. Springer, 2019, pp. 3–10.

[3] J. S. Notland, J. Li, M. Nowostawski, and P. H. Haro, "Sok: Cross-chain bridging architectural design flaws and mitigations," 2024. [Online]. Available: https://arxiv.org/abs/2403.00405

[4] N. Li, M. Qi, Z. Xu, X. Zhu, W. Zhou, S. Wen, and Y. Xiang, "Blockchain cross-chain bridge security: Challenges, solutions, and future outlook," *Distrib. Ledger Technol.*, Oct. 2024, just Accepted. [Online]. Available: https://doi.org/10.1145/3696429

[5] A. Augusto, R. Belchior, J. Pfannschmidt, A. Vasconcelos, and M. Correia, "Xchainwatcher: Monitoring and identifying attacks in cross-chain bridges," 2024. [Online]. Available: https://arxiv.org/abs/2410.02029

[6] P. Cichonski, T. Millar, T. Grance, K. Scarfone *et al.*, "Computer security incident handling guide," *NIST Special Publication*, vol. 800, no. 61, pp. 1–147, 2012.

[7] O. Rikken, M. Janssen, Z. Kwee, R. Bolívar, and H. Scholl, "Governance challenges of blockchain and decentralized autonomous organizations," *Info. Pol.*, vol. 24, no. 4, p. 397–417, Jan. 2019. [Online]. Available: https://doi.org/10.3233/IP-190154

[8] T. Hardjono, M. Hargreaves, N. Smith, and V. Ramakrishna, "Secure Asset Transfer (SAT) Interoperability Architecture," Internet Engineering Task Force, Internet-Draft draft-ietf-satp-architecture-06, Dec. 2024, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-satp-architecture/06/

[9] M. Hargreaves, T. Hardjono, R. Belchior, and V. Ramakrishna, "Secure Asset Transfer Protocol (SATP) Core," Internet Engineering Task Force, Internet-Draft draft-ietf-satp-core-07, Oct. 2024, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-satp-core/07/

[10] R. Belchior, M. Correia, A. Augusto, and T. Hardjono, "Secure Asset Transfer Protocol (SATP) Gateway Crash Recovery Mechanism," Internet Engineering Task Force, Internet-Draft draft-belchior-satp-gateway-recovery-03, Jan. 2025, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-belchior-satp-gateway-recovery/03/

[11] R. Belchior, P. Somogyvari, J. Pfannschmidt, A. Vasconcelos, and M. Correia, "Hephaestus: Modeling, analysis, and performance evaluation of cross-chain transactions," *IEEE Transactions on Reliability*, vol. 73, no. 2, pp. 1132–1146, Jun. 2024, publisher Copyright: © 1963-2012 IEEE.

[12] E. Abebe, P. Robinson, A. Chand, M. Murdock, and D. Hyland-Wood, "Crosschain Risk Framework," Oct. 2023, accessed: 2024-11-11. [Online]. Available: https://crosschainriskframework.github.io/

[13] M. Zhang, X. Zhang, J. Barbee, Y. Zhang, and Z. Lin, "Sok: Security of cross-chain bridges: Attack surfaces, defenses, and open problems," 2023. [Online]. Available: https://arxiv.org/abs/2312.12573

[14] S.-S. Lee, A. Murashkin, M. Derka, and J. Gorzny, "Sok: Not quite water under the bridge: Review of cross-chain bridge hacks," 2022. [Online]. Available: https://arxiv.org/abs/2210.16209

[15] N. Belenkov, V. Callens, A. Murashkin, K. Bak, M. Derka, J. Gorzny, and S.-S. Lee, "Sok: A review of cross-chain bridge hacks in 2023," 2025. [Online]. Available: https://arxiv.org/abs/2501.03423

[16] N. Brownlee and E. Guttman, "Expectations for computer security incident response," Tech. Rep., 1998.

[17] M. J. West-Brown, D. Stikvoort, K.-P. Kossakowski, G. Killcrece, R. Ruefle, and M. Zajicek, *Handbook for computer security incident response teams (CSIRTs)*. Carnegie Mellon University, Software Engineering Institute, 1998.

[18] C. Alberts, A. Dorofee, G. Killcrece, R. Ruefle, and M. Zajicek, "Defining incident management processes for csirts: A work in progress," *SEI. Octubre*, 2004.