G-Prove: Gossip-Based Provenance for Scalable Detection of Cross-Domain Flow Attacks in SDN

Moustapha Awwalou Diouf
SnT, University of Luxembourg
moustapha.diouf@uni.lu

Maimouna Tamah Diao
SnT, University of Luxembourg
maimouna.diao@uni.lu

El-hacen Diallo
SnT, University of Luxembourg
el-hacen.diallo@uni.lu

Samuel Ouya
Cheikh Hamidou KANE Digital University
samuel.ouya@unchk.edu.sn

Jacques Klein
SnT, University of Luxembourg
jacques.klein@uni.lu

Tegawendé F. Bissyandé
SnT, University of Luxembourg
tegawende.bissyande@uni.lu

***Abstract*—Software-defined networking (SDN) is widely adopted in enterprise networks, data centers, and wide-area networks. These infrastructures are often federated into multiple administrative domains managed by distinct organizations. In this context, forensic analysis of cross-domain attacks remains a major challenge: fragmented causal visibility across domains and privacy constraints prevent effective tracing of threat propagation. Although prior work has focused on centralized provenance systems offering causal traceability, these approaches do not scale in multi-domain contexts with heterogeneous policies. We propose G-Prove, a decentralized forensic framework for multi-domain SDN environments. G-Prove builds local provenance graphs and anchors cross-domain events via a cryptographically signed DAG, enabling causal analysis without exposing each domain's internal data. Our results on a cross-domain attack scenario demonstrate the feasibility and effectiveness of G-Prove, and allow us to identify areas for improvement for more complex deployments.**

I. Introduction

Over the past decade, Software-Defined Networking (SDN) has revolutionized the way networks are designed, managed, and secured. By decoupling the control and data planes, SDN enables centralized programmability, dynamic policy enforcement, and granular traffic control. Early deployments, such as Google's B4 [1] and Microsoft's SWAN [2], demonstrated SDN's potential for scalability and flexibility. Commercial solutions like VMware NSX [3] and Cisco ACI [4] have accelerated its adoption, and by 2024, more than 33% of enterprise LAN/WAN/SD-WAN segments are managed via cloud-hosted SDN controllers, with 72% of enterprises planning unified, multi-domain platforms by 2026 [5].

With the relentless expansion of network infrastructures, the performance of a single controller has emerged as a limitation for network progression. This constraint stems from inherent restrictions in computing power, storage capacity, and network bandwidth allocated to a single controller. To address these challenges, a multi-controller architecture has been introduced [6], offering advantages in terms of flexibility and scalability. In a multi-domain SDN network, different administrative or geographically distributed domains collaborate to provide integrated services. This shift introduces new challenges as each domain may have security and privacy concerns regarding the disclosure of its information to other domains.

A critical dimension of SDN security lies in its event-driven architecture. Modern SDN controllers, such as ONOS, OpenDaylight, and Floodlight, react to network changes by dispatching events to modular applications [7]. The rapid deployment of SDN applications by multiple entities leads to complex network environments, vulnerable to various cross-plane attacks. These attacks can occur when an application ignores or inconsistently processes received events [7]. More subtly, when the controller silently updates its internal state without generating expected events, thereby preventing downstream applications from reacting [8]. These event-based cross-plane attacks are particularly dangerous in multi-domain SDN environments. They fall under logical bugs requiring both local and global understanding of events [7], making them difficult to detect automatically. In such contexts, traditional monitoring tools are insufficient: they cannot trace causal relationships between control and data plane events cross-domains, hindering forensic investigations and accountability [9].

Despite extensive research on cross-plane vulnerabilities, a common limitation emerges: these works focus exclusively on centralized provenance-based frameworks [8], [10], [11], [12] without considering the multi-domain case. Among provenance-based solutions, PICOSDN [8] uses causal provenance graphs to trace interactions between the control and data planes within a single SDN domain, helping operators understand how specific control decisions were made. PROVGUARD [11] enforces information flow policies by monitoring the provenance of control actions at runtime and detecting policy manipulation attacks. Although these systems offer strong accountability and policy enforcement, they assume a

Workshop on Attack Provenance, Reasoning, and Investigation for Security in the Monitored Environment (PRISM) 2026
23 February 2026, San Diego, CA, USA
ISBN 978-1-970672-00-8
https://dx.doi.org/10.14722/prism.2026.23026
www.ndss-symposium.org

centralized and trusted control plane. These assumptions do not hold in multi-domain SDN environments, where privacy policies differ, and controllers operate independently without sharing the complete network state.

In this paper, we present G-Prove, a decentralized forensic framework that combines provenance-based causal analysis with cross-domain coordination. Each domain deploys a Trusted Provenance Emitter (TPE) that locally collects Data Plane and Control Plane events, modeling them using the W3C PROV-DM standard. Instead of broadcasting complete provenance graphs, domains submit signed Merkle roots to a shared, append-only DAG. This DAG is propagated via a gossip protocol, enabling asynchronous and scalable dissemination while preserving confidentiality.

We make two major contributions:

- We propose G-Prove, a novel decentralized forensic architecture that enables cross-domain accountability while preserving each domain's confidentiality by sharing only cryptographic digests of local provenance data.
- A gossip-based DAG synchronization protocol with causal ordering that achieves consistency without global coordination.

We demonstrate the feasibility of G-Prove through an attack scenario on a two-domain prototype. We also discuss current limitations and directions for extending the framework to larger deployments.

Data Availability. The implementation of our approach and the experimental data are available online [13].

II. Background

A. Multi-domain in Software Defined Networking

SDN offers flexibility, abstraction, programmability, and virtualization to overcome the gaps and disadvantages of traditional network architecture [14]. The Open Networking Foundation [15] conceptualizes the SDN architecture in three distinct layers, as illustrated in figure 1. Here, the infrastructure layer refers to the data plane. To improve the programmability of the control plane, it is divided into two distinct layers: the control layer and the application layer.

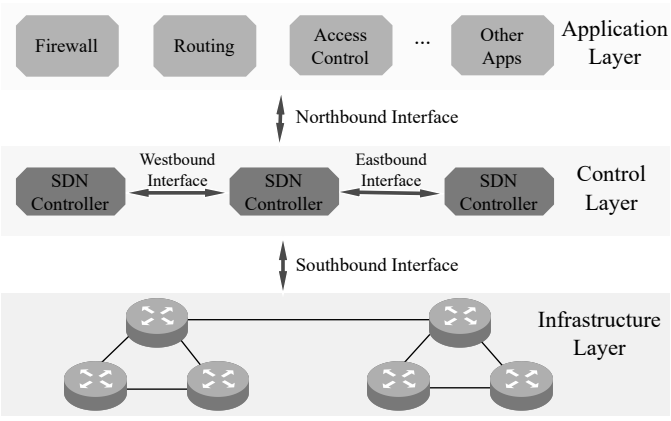

Fig. 1. SDN Architecture.

Application layer. The layer includes applications that take advantage of SDN-enabled services such as firewall, routing, access control, etc, provided by the control layer, and the two layers interact via the northbound interface [16], [17].

Control layer. This layer contains one or more controllers, known as the "brain of SDN," that implement control logic and expose programming interfaces for network flexibility [18].

Infrastructure layer. This layer includes devices that forward packets based on controller instructions. Data plane devices leverage the southbound interface to communicate with controllers, such as the well-known OpenFlow protocol [19].

Multi-domain SDN appears as one of the solutions for implementing SDN in large-scale networks [6]. Controllers communicate via East/Westbound interfaces to coordinate cross-domain operations, as illustrated in Figure 1. However, these interfaces are not yet standardized [6].

B. Security

SDN programmability significantly modifies the attack surface of the control plane. Prior work [7], [20], [10], [21] has demonstrated cross-plane attacks, which allow attackers to influence control plane decision-making without directly attacking the controller. A prominent example of such a vulnerability is CVE-2018-12691 [7], a logical bug that prevents the ONOS firewall from installing flow rules, thereby allowing unauthorized communication despite restrictive security policies. The impact of such attacks is amplified in multi-domain environments. These vulnerabilities are difficult to detect as they require a deep causal understanding of events across administrative boundaries, a capability lacking in traditional monitoring tools.

C. Data Provenance

Data provenance refers to the complete record of data history: its origin, transformations, and the actors involved. This concept addresses the fundamental questions "***who, what, when, where***" about data, enabling the detection of falsification and reinforcing trust [22]. The World Wide Web Consortium (W3C) defines a standard provenance model, PROV-DM [23], based on three fundamental elements:

- **Entity:** a piece of data or an object, such as a network packet or a flow rule.
- **Activity:** an action that modifies or generates an entity, such as packet transmission.
- **Agent:** an actor responsible for an activity, such as a host, switch, or controller.

These elements and their causal relationships (such as *used*, *wasAttributedTo*, and *wasAssociatedWith*) form a provenance graph that documents data lineage. The graph is constructed forward through events and analyzed backward to trace origins.

III. Threat Model

We consider a multi-domain SDN architecture where each administrative domain is managed by its own SDN controller and equipped with a Trusted Provenance Emitter (TPE). While

the framework is designed for a general multi-domain context, our model and evaluation focus on a representative two-domain scenario (D_1 and D_2).

Assumptions. The TPEs are part of the trusted computing base and operate outside the critical data path. Communication between TPEs via the gossip protocol is authenticated and integrity-protected. Once recorded in the DAG, provenance entries are protected against retroactive modification through Merkle hashing.

Attacker. We consider that an attacker can inject falsified information to mislead the SDN controller. The attacker may also exploit controller APIs to inject malicious flow rules, including through compromised applications. A compromised controller cannot falsify provenance records. The attacker cannot compromise the TPE or tamper with the gossip channel.

Security Properties. Against this attacker, G-Prove ensures *non-repudiation* via digital signatures (σ_D), *tamper-evidence* through cryptographic chaining ($H(e^D_{i-1})$), and *confidentiality* of internal domain data by sharing only Merkle digests inter-domain.

IV. Approach

G-Prove is a scalable, accountable, and privacy-preserving forensic architecture for multi-domain SDN environments. Our design introduces a gossip-propagated DAG structure to anchor cross-domain provenance, enabling decentralized coordination without global consensus.

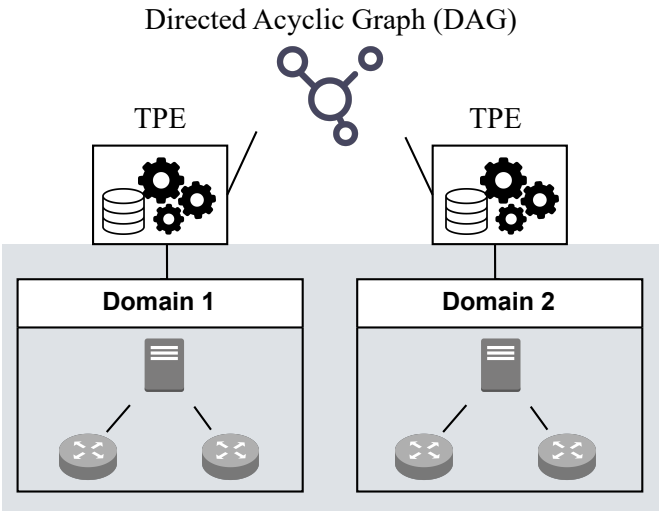

Fig. 2. G-Prove Architecture.

System Overview. As illustrated in Figure 2, the architecture is composed of two layers:

- *Local SDN Layer.* Each domain deploys a Trusted Provenance Emitter (TPE) that collects cross-domain events and maintains a local provenance graph following the PROV-DM model defined in Section II-C. Events are recorded locally and not exposed externally unless they affect cross-domain behavior.

Algorithm 1 DAG Entry Commitment & Propagation

Require: Local provenance records R, Previous entry e^D_{i-1}, Domain key K_D
Ensure: New DAG entry e^D_i propagated to peers
1: $M_i \leftarrow \text{MERKLEROOT}(R)$
2: $t_i \leftarrow \text{INCREMENTLAMPORTCLOCK}()$
3: $h_{prev} \leftarrow H(e^D_{i-1})$
4: $payload \leftarrow (M_i, t_i, D, h_{prev})$
5: $\sigma_D \leftarrow \text{SIGN}(payload, K_D)$
6: $e^D_i \leftarrow (payload, \sigma_D)$
7: $\text{GOSSIP}(e^D_i)$

- *Cross-domain DAG Layer.* TPEs submit signed provenance digests to a distributed DAG, synchronized via a gossip protocol without global coordination.

DAG Structure and Propagation. Each DAG entry is defined as:

$$e^D_i = (M_i, t_i, D, H(e^D_{i-1}), \sigma_D)$$

Where:

- M_i is the Merkle root of the interdomain provenance records.
- t_i is the Lamport logical timestamp used to establish causal ordering, updated as $t = \max(t, t_{received}) + 1$.
- D is the unique domain identifier.
- $H(e^D_{i-1})$ is the cryptographic hash of the domain's previous entry, ensuring acyclicity and chronological chaining.
- σ_D is the digital signature of the domain, providing authenticity and non-repudiation.

This structure ensures tamper-evidence: modifying any entry invalidates all subsequent entries from that domain. Domains synchronize via gossip, exchanging missing entries until convergence. Since entries from different domains do not conflict, they are merged without consensus.

Commitment Process. When cross-domain events occur, the TPE collects the corresponding local provenance records and computes a Merkle tree over them. The resulting root (M_i) is included in a new DAG entry, which is signed (σ_D) and chained to the domain's previous entry via $H(e^D_{i-1})$ before propagation. The complete commitment and propagation procedure is summarized in Algorithm 1.

Cross-domain Scope. DAG writes are triggered only for events affecting cross-domain communication. Intra-domain events are recorded locally but do not generate DAG entries.

Confidentiality. While provenance graphs enhance accountability, they may expose sensitive operational details such as host behaviors or internal policies. To preserve confidentiality, full graphs are never published. Instead, each domain submits only a signed Merkle root to the shared DAG. Host bindings shared via gossip are hashed (SHA-256) to enable cross-domain correlation without exposing plaintext addresses. Intra-domain events remain strictly local.

Forensic Analysis. G-Prove adopts a post-hoc validation model. Network operations proceed without blocking; provenance is recorded asynchronously. When an incident occurs, investigators query local provenance graphs and use the DAG

to correlate events across domains. The signed entries provide a verifiable timeline of each domain's actions.

V. Evaluation

We evaluate G-Prove on a two-domain SDN testbed to answer two questions: (1) Can G-Prove identify attackers in scenarios that require cross-domain correlation? (2) Does the gossip protocol achieve DAG consistency while preserving confidentiality?

A. Experimental Setup

We emulate a two-domain SDN environment using Mininet (v2.3.5) with two ONOS controllers (v2.5.1) running on separate virtual machines. Each domain contains two Open vSwitch switches and two hosts. TPEs are implemented in Python using Scapy for packet capture and SQLite for local storage. Gossip uses TCP with random peer selection. Host bindings are hashed using SHA-256 before sharing via gossip to preserve confidentiality. Source code is available.[1]

B. Attack Scenario

Inspired by the cross-plane vulnerabilities identified by Ujcich et al. [7], we evaluate G-Prove against a cross-domain IP spoofing attack. An attacker in D_2 generates ICMP traffic using the source IP address of a victim host in D_1. Because SDN controllers lack standardized inter-domain interfaces for real-time verification of host-to-IP bindings, the controller in D_1 cannot determine whether this traffic originates from a legitimate internal host or a foreign entity.

C. Results

Figure 3 shows the provenance graph captured during the attack.

Cross-Domain Correlation. In D_1, the TPE records a packet attributed to a host with H(MAC) `da9098...` and H(IP) `a9a512...`. Without cross-domain information, D_1 observes an unknown MAC but cannot determine its origin. Through the gossip protocol, D_1 receives a binding from D_2 associating the same H(MAC) with H(IP) `180d33...` (Table I).

TABLE I
Cross-Domain Correlation

Source	H(MAC)	H(IP)
D_1 observed	da9098...	a9a512...
D_2 binding (gossip)	da9098...	180d33...

The matching H(MAC) confirms the observed host belongs to D_2. The differing H(IP) values (`a9a512...` $\neq$ `180d33...`) confirm IP spoofing. G-Prove identifies the attacker as a D_2 entity—an identification that would be impossible without cross-domain correlation.

DAG Consistency and Confidentiality. After gossip synchronization, both domains hold identical DAG entries, demonstrating consistency without global coordination. Each entry is signed and hash-chained, ensuring tamper-evidence.

[1]https://anonymous.4open.science/r/G-Prove-D1C7

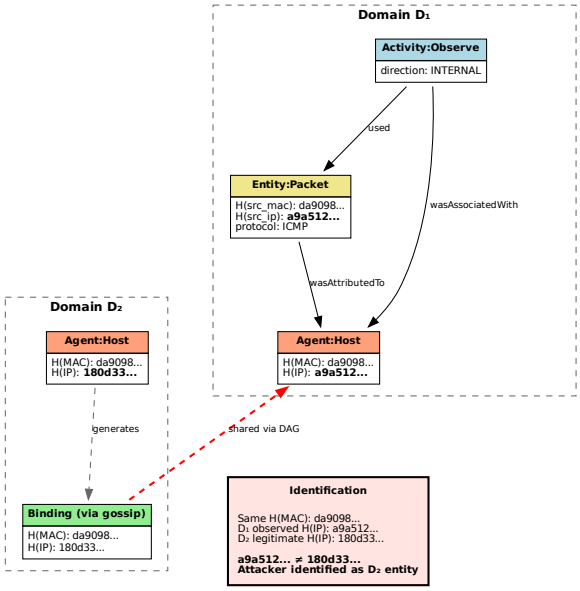

Fig. 3. Provenance graph of the cross-domain attack. D_1 observes H(MAC) `da9098...` with H(IP) `a9a512...`. The binding from D_2 associates the same H(MAC) with H(IP) `180d33...`, enabling identification.

Table II shows that identification is achieved while preserving domain confidentiality. Only SHA-256 hashes are shared via the gossip protocol; plaintext MAC and IP addresses never leave their respective domains.

TABLE II
Information Shared via Gossip

Information	Shared?
Attacker's domain	✓
H(MAC), H(IP)	✓
Plaintext MAC/IP	—
Internal topology	—

D_1 identifies the attacker's origin without learning D_2's internal addresses. The one-way property of SHA-256 ensures that shared hashes cannot be reversed to recover original values.

D. Performance Metrics

To quantify G-Prove's practical overhead, we measure five metrics, namely audit latency, attribution effectiveness, storage growth, bandwidth consumption, and time-to-consistency (Table III).

Audit Latency. Cross-domain correlation completes in under 1 ms (0.87 ms for data loading, 0.03 ms for analysis).

Attribution Effectiveness. G-Prove correctly identifies the attacker as a foreign domain entity.

Storage Growth. Each DAG entry averages 264 bytes, with the database totaling 114.7 KB for 202 provenance nodes.

TABLE III
G-PROVE PERFORMANCE METRICS

Category	Metric	Value
Audit Latency	Data loading	0.87 ms
	Attack analysis	0.03 ms
Attribution Effectiveness	Attribution rate	100%
Storage Growth	Database size	114.7 KB
	Avg bytes per DAG entry	264 bytes
Bandwidth Consumption	Bytes sent	13.56 KB
	Bytes received	12.98 KB
Time-to-Consistency	Avg propagation delay	2.31 ms
	Min / Max delay	1.64 / 3.46 ms

Bandwidth Consumption. The gossip protocol exchanges approximately 26.5 KB (13.56 KB sent, 12.98 KB received) to synchronize DAG entries across domains.

Time-to-Consistency. DAG entries propagate with an average delay of 2.31 ms (min: 1.64 ms, max: 3.46 ms).

VI. DISCUSSION

This work explores the feasibility of cross-domain forensic analysis in multi-domain SDN environments. Our results demonstrate that localized decisions based on incomplete visibility can lead to global security failures—a gap that G-Prove addresses through gossip-based DAG synchronization and hash-protected bindings.

Design Trade-offs. G-Prove's architecture reflects deliberate trade-offs compared to alternative coordination approaches. A centralized coordinator would simplify orchestration but introduces a single point of failure and requires all domains to trust a central authority [24]. Blockchain-based ledgers eliminate central trust but incur significant overhead, as consensus protocols such as PBFT require $O(N^2)$ message complexity [25]; moreover, the shared ledger remains visible to all participants, raising confidentiality concerns [26]. CT-style append-only logs provide public verifiability but similarly lack confidentiality since entries are publicly accessible.

G-Prove avoids these limitations by combining gossip-based synchronization with cryptographic digests. The gossip protocol achieves convergence with linear message complexity and non-blocking updates [27], providing robustness without central coordination. Confidentiality is preserved because full provenance graphs are never published; only signed Merkle roots are shared, and host bindings are hashed before propagation. The trade-off is eventual consistency rather than strong consistency—acceptable for post-hoc forensic analysis where immediate global agreement is not required.

Research Outlook. We envision G-Prove as a foundation for cross-domain accountability in programmable networks. We plan to extend G-Prove to Intent-Based Networking (IBN) environments, where high-level policies are translated into network configurations across multiple domains. In such contexts, provenance must capture not only data plane events but also intent-to-configuration mappings, enabling investigators to trace policy violations back to their source. We also intend to explore the use of Large Language Models (LLMs) for automated provenance graph analysis. LLMs could assist investigators by querying complex graphs in natural language, identifying anomalous patterns, and generating human-readable explanations of attack causality.

VII. RELATED WORK

There is a lack of research in provenance-based SDN security approaches. Moreover, most existing research only considers single-domain SDN environments, leaving a notable gap in multi-domain SDN contexts.

ForenGuard [12] proposes a provenance-based root cause analysis framework that records event propagation from the data plane to the control plane. ProvSDN [28] aims to locate the root cause of cross-app poisoning attacks by backtracing corrupted data. PicoSDN [8] extends this work using both data plane and control plane attributes with fine-grained partitioning to improve provenance graph interpretability. G-Prove differs by introducing a gossip-based shared DAG, enabling detection of attacks that require cross-domain correlation.

VIII. CONCLUSION

In this paper, we presented G-Prove, a decentralized forensic framework for multi-domain SDN environments. Unlike prior work, which focuses on single-domain provenance, G-Prove enables cross-domain accountability by anchoring local provenance graphs to a shared DAG synchronized via gossip protocol. Our evaluation on a cross-domain IP spoofing attack demonstrates that G-Prove can identify attackers that would remain undetectable without cross-domain correlation, while preserving confidentiality through hash-based bindings.

Future Work. We identify several directions for future research. First, we plan to evaluate G-Prove on larger topologies with more domains to assess scalability. Second, we aim to extend the framework to Intent-Based Networking environments. Third, we intend to explore LLM-assisted provenance graph analysis for automated forensic reasoning.

REFERENCES

[1] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat, "B4: experience with a globally-deployed software defined wan," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 3–14. [Online]. Available: https://doi.org/10.1145/2486001.2486019

[2] C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer, "Achieving high utilization with software-driven wan," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, p. 15–26, Aug. 2013. [Online]. Available: https://doi.org/10.1145/2534169.2486012

[3] VMware, Inc., "VMware NSX network virtualization platform," https://www.vmware.com/products/cloud-infrastructure/nsx, accessed: 2025.

[4] Cisco Systems, Inc., "Cisco Application Centric Infrastructure," https://www.cisco.com/go/aci, accessed: 2025.

[5] ——, "2024 Global Networking Trends Report," https://www.cisco.com/c/dam/global/en_uk/solutions/enterprise-networks/2024-global-networking-trends.pdf, January 2024, accessed: 2025-07-04, see page 6.

[6] F. X. Wibowo, M. A. Gregory, K. Ahmed, and K. M. Gomez, "Multi-domain software defined networking: Research status and challenges," *Journal of Network and Computer Applications*, vol. 87, pp. 32–45, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804517300991

[7] B. E. Ujcich, S. Jero, R. Skowyra, S. R. Gomez, A. Bates, W. H. Sanders, and H. Okhravi, "Automated discovery of cross-plane event-based vulnerabilities in software-defined networking," in *Network and Distributed System Security Symposium*, 2020.

[8] B. E. Ujcich, S. Jero, R. Skowyra, A. Bates, W. H. Sanders, and H. Okhravi, "Causal analysis for {Software-Defined} networking attacks," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 3183–3200.

[9] A. H. Abdi, L. Audah, A. Salh, M. A. Alhartomi, H. Rasheed, S. Ahmed, and A. Tahir, "Security control and data planes of sdn: A comprehensive review of traditional, ai, and mtd approaches to security solutions," *IEEE Access*, vol. 12, pp. 69 941–69 980, 2024.

[10] S. Hong, L. Xu, H. Wang, and G. Gu, "Poisoning network visibility in software-defined networks: New attacks and countermeasures." in *Ndss*, vol. 15, 2015, pp. 8–11.

[11] Z. Liu, J. Mao, J. Zeng, J. Li, Q. Lin, J. Liu, J. Zhuge, and Z. Liang, "Provguard: Detecting sdn control policy manipulation via contextual semantics of provenance graphs," in *NDSS*, 2025.

[12] H. Wang, G. Yang, P. Chinprutthiwong, L. Xu, Y. Zhang, and G. Gu, "Towards fine-grained network security forensics and diagnosis in the sdn era," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 3–16. [Online]. Available: https://doi-org.proxy.bnl.lu/10.1145/3243734.3243749

[13] Moustapha. (2025, Sep.) G-prove. [Online]. Available: https://anonymous.4open.science/r/G-Prove-D1C7

[14] J. Singh and S. Behal, "Detection and mitigation of ddos attacks in sdn: A comprehensive review, research challenges and future directions," *Computer Science Review*, vol. 37, p. 100279, 2020.

[15] O. N. F. ONF. (2017, Sep.) Software-defined networking (sdn) definition. [Online]. Available: https://opennetworking.org/

[16] B. Rauf, H. Abbas, M. Usman, T. A. Zia, W. Iqbal, Y. Abbas, and H. Afzal, "Application threats to exploit northbound interface vulnerabilities in software defined networks," *ACM Comput. Surv.*, vol. 54, no. 6, July 2021. [Online]. Available: https://doi.org/10.1145/3453648

[17] M. A. Diouf, S. Ouya, J. Klein, and T. F. Bissyandé, "Software security in software-defined networking: A systematic literature review," *arXiv preprint arXiv:2502.13828*, 2025.

[18] B. Almadani, A. Beg, and A. Mahmoud, "Dsf: A distributed sdn control plane framework for the east/west interface," *IEEE Access*, vol. 9, pp. 26 735–26 754, 2021.

[19] A. Liatifis, P. Sarigiannidis, V. Argyriou, and T. Lagkas, "Advancing sdn from openflow to p4: A survey," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–37, 2023.

[20] M. Dhawan, R. Poddar, K. Mahajan, and V. Mann, "Sphinx: detecting security attacks in software-defined networks." in *Ndss*, vol. 15, 2015, pp. 8–11.

[21] R. Skowyra, L. Xu, G. Gu, V. Dedhia, T. Hobson, H. Okhravi, and J. Landry, "Effective topology tampering attacks and defenses in software-defined networks," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2018, pp. 374–385.

[22] F. T. Jaigirdar, C. Rudolph, and C. Bain, "Prov-iot: A security-aware iot provenance model," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020, pp. 1360–1367.

[23] K. Belhajjame, R. B'Far, J. Cheney, S. Coppens, S. Cresswell, Y. Gil, P. Groth, G. Klyne, T. Lebo, J. McCusker *et al.*, "Prov-dm: The prov data model," *W3C Recommendation*, vol. 14, pp. 15–16, 2013.

[24] G. He, W. Su, S. Gao, N. Liu, and S. K. Das, "NetChain: A blockchain-enabled privacy-preserving multi-domain network slice orchestration architecture," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 188–202, 2022.

[25] M. Xu, C. Wang, Y. Zou, D. Yu, X. Cheng, and W. Lv, "Curb: Trusted and scalable software-defined network control plane for edge computing," in *Proc. IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*, 2022.

[26] Y. Liu, W. Yang, Y. Wang, and Y. Liu, "An access control model for data security sharing cross-domain in consortium blockchain," *IET Blockchain*, vol. 3, pp. 18–34, 2023.

[27] E. Sakic and W. Kellerer, "Impact of adaptive consistency on distributed SDN applications: An empirical study," *IEEE Transactions on Network and Service Management*, 2020.

[28] B. E. Ujcich, S. Jero, A. Edmundson, Q. Wang, R. Skowyra, J. Landry, A. Bates, W. H. Sanders, C. Nita-Rotaru, and H. Okhravi, "Cross-app poisoning in software-defined networking," in *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, 2018, pp. 648–663.