A Context-Aware Access Control Scheme for Beyond Zero-Trust in the SSH Protocol

*Sewoong Jeong, †Mario Cutillas, *Samariddin, ‡Utkarsha Kshirsagar, *Jaehoon (Paul) Jeong, and ‡Tae (Tom) Oh
*Sungkyunkwan University, †University of Alicante, ‡Rochester Institute of Technology
Email: {jsw1301, smavlitdinov, pauljeong}@skku.edu, mcp173@alu.ua.es, {uk9263, thoics}@rit.edu

***Abstract*—Traditional mandatory Virtual Private Network (VPN) tunneling for SSH access causes significant performance degradation in practice. Leveraging the capabilities of Software-Defined Networking (SDN), this paper proposes a context-aware access control framework for SSH authentication that integrates multiple security-relevant contexts validated in prior work, including MAC and IP addresses, subnet category, geolocation, time-of-day, and historical access information. The framework employs RESTCONF, OpenDaylight controller, and OpenFlow protocol to support standard-compliant and programmable security operations. It utilizes gathered context features and classifies each connection attempt into trusted or untrusted. Classification results are used to determine whether traffic should use a direct path or a VPN tunnel. Our experiment demonstrates that modulating the number of VPN-routed sessions improves throughput and reduces latency. These results show that selective VPN tunneling effectively alleviates the systematic performance penalty inherent in traditional and universal methods.**

***Index Terms*—Software-Defined Networking, Secure Shell, Context-Aware, Access Control, Virtual Private Network.**

I. Introduction

Secure Shell (SSH) connections are often combined with VPN-based additional authentication to establish a layered defense. However, indiscriminate tunneling often acts as a possible bottleneck of the system, and results in performance degradation [1]. To address this issue, we propose context-aware access control for SSH connections within an SDN environment. Getting rid of a rigid policy, this framework assigns a risk level to each connection after considering the values of predefined security attributes. This enforces VPN path only for traffic that the SDN controller considers suspicious, while allowing trusted traffic to follow direct paths. Such an adaptive mechanism enables simultaneous improvements of security and performance.

II. Design

The proposed system, as shown in Fig. 1, includes the context collector which gathers attributes from each login attempt when an SSH connection is initiated. Gathered information includes network identifiers (IP/MAC), subnet category, geolocation [2], time-of-day [3], and history [3] information. Next, these attributes are securely transmitted to the SDN controller using RESTCONF. Within the controller, risk engine calculates a risk score of current attempt, using information stored in the internal DB. The engine learns from this information and dynamically makes a decision for each attempt.

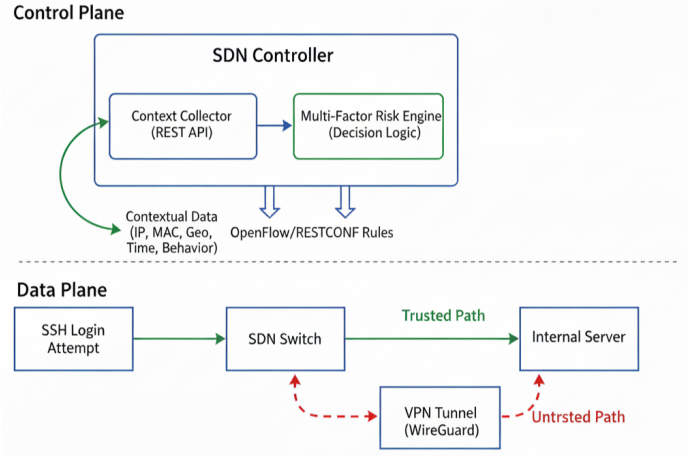


Fig. 1. A Framework of Context-Aware Access Control System.

The controller employs a multi-factor risk evaluation engine to compute a composite risk score. A risk score is calculated by assessing the deviation from the behavioral baseline, which is expected by an engine. Trustworthiness of IP and MAC address pair, consistency with past behavioral features, subnet familiarity, and geographic irregularity are all considered to make a decision. Based on the predefined threshold, each connection attempt is classified into trusted or untrusted.

For a trusted connection, the controller minimizes latency and unnecessary tunnel overhead by opening a direct path to the server. However, for an untrusted connection, the controller leverages step-up defense by redirecting the traffic to a VPN tunnel. This additional security process is activated only when a high-risk attempt is detected. This design enables automated, consistent, and scalable access control in SDN environments, while selectively enforcing VPN tunneling for both security and performance.

III. Emulation

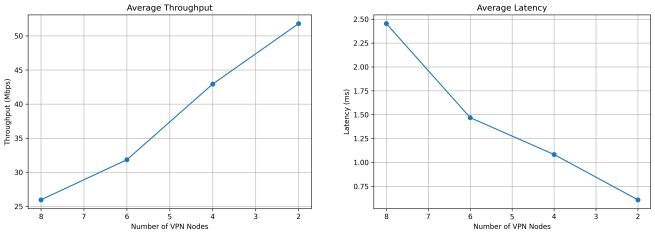
A. Emulation Setup

We evaluated the performance impact of selective VPN enforcement using an SDN emulation built with Mininet [4]

TABLE I
WEIGHTED AVERAGE PERFORMANCE METRIC AND NOTATION.

$$\text{Average} = D \cdot \frac{N_{\text{direct}}}{N_{\text{total}}} + V \cdot \frac{N_{\text{vpn}}}{N_{\text{total}}}$$

Symbol	Description
D	Direct path performance metric (latency or throughput)
V	VPN path performance metric (latency or throughput)
N	The number of clients



(a) Throughput (b) Latency

Fig. 2. Performance Impact of Selective VPN Enforcement.

and the OpenDaylight [5] controller. The emulated topology consists of eight client hosts, three OpenFlow switches, and a single SSH server. To model realistic access-network constraints, a bandwidth bottleneck of 100 Mbps was imposed on the server-facing links using Mininet's traffic control mechanisms.

VPN tunnels are implemented using WireGuard [1], and selectively applied between the client side and the server side. To emulate different policies, the portion of VPN clients varies with four specific configurations (i.e., 100%, 75%, 50%, and 25%). Each case was repeated 30 times to ensure that observations follow a normal distribution.

Latency was measured using the average ICMP round-trip time (RTT) as a proxy for SSH responsiveness, while throughput was measured using TCP iperf3 sessions initiated through SSH to emulate authenticated access workflows. For each experiment, the performance of the direct path and the VPN path is measured separately, and overall throughput and latency are calculated as a weighted average, whose weight is based on the portion of VPN-routed and direct client.

B. Results

We evaluated how the portion of VPN-routed SSH session affects network performance. This emulation highlights the costs of enforcing VPN routing and confirms that our context-aware controller should enforce VPN tunneling only when necessary.

As shown in Fig. 2(a), throughput was the lowest at 26 Mbps when all clients were indiscriminately routed to the VPN tunnel. Lowering VPN portion improved throughput, while throughputs were 31 Mbps, 43 Mbps, and 52 Mbps when the portion of VPN-routed clients were 75%, 50%, and 25%, respectively. This result demonstrates that mandatory tunneling imposes substantial performance degradation, while selective VPN routing alleviates tunnel contention.

Latency results, illustrated in Fig. 2(b), show a consistent downward trend as fewer hosts participate in the VPN tunnel. With all clients routed through the VPN, the measured latency reached 2.5 ms, but it dropped to 1.5 ms, 1.1 ms, and 0.5 ms as the VPN proportion decreased to 75%, 50%, and 25%, respectively. This relationship indicates that VPN congestion increases queuing delay within the encrypted tunnel, amplifying end-to-end latency.

The experiments confirm that context-aware control can effectively address the trade-off between network security and performance. Compulsory VPN tunneling does act as a performance bottleneck due to network contention and latency. However, selective VPN tunneling ensures a secure network by blocking high-risk connection attempts, without degrading user experience.

The source code for implementation and experiments is available at our GitHub repository: https://github.com/jaehoonpauljeong/KICS2026-Group6.

IV. CONCLUSION

Zero-trust authentication has been widely adopted as a highly secure model in countless real-world applications. However, there is a performance degradation issue when multiple protocols are combined for better security. Especially when SSH and VPN are utilized together, our results show that blanket enforcement is not optimal for network environments, and that selectively forcing VPN tunneling for better throughput and lower latency, which are critical to user experience.

Our future work aims to work on larger network topologies that mimic real-world network environments. This includes increasing the number of nodes and the capacity of each link. In addition, there should be a more precise policy and broader coverage for various secure connection protocols.

ACKNOWLEDGMENTS

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant (No. RS-2024-00398199) and the National Research Foundation of Korea (NRF) grant (No. 2023R1A2C2002990) funded by the Ministry of Science and ICT (MSIT), South Korea. Jaehoon (Paul) Jeong is the corresponding author.

REFERENCES

[1] M. Pudelko, P. Emmerich, S. Gallenmüller, and G. Carle, "Performance Analysis of VPN Gateways," in *2020 IFIP Networking Conference (Networking)*, 2020, pp. 325–333.

[2] G. Pavlov and N. Tagarev, "Enhancing Cybersecurity through the Integration of Geographic Information Systems Technologies," *International Journal of Science and Research (IJSR)*, vol. 13, 09 2024.

[3] H. Sharma, "Behavioral Analytics and Zero Trust," *International Journal of Information Technology and Management Information Systems (IJITMIS)*, vol. 12, no. 1, pp. 63–84, 2021.

[4] Mininet, "Mininet: An Instant Virtual Network on your Laptop (or other PC)," 2026, [Online]. Available: http://mininet.org/.

[5] OpenDaylight, "OpenDaylight SDN Controller," 2026, [Online]. Available: https://www.opendaylight.org/.