Poster: Practical Swarm Attestation using Probabilistic Program Integrity Verification

Sanket C Uppin, Mehreen Jabbeen, Rijurekha Sen, Vireshwar Kumar
Indian Institute of Technology, Delhi
sanket@sit.iitd.ac.in, {mehreen.jabbeen, riju, viresh}@cse.iitd.ac.in

***Abstract*—There is an increasing trend to operate vehicles (automobiles and drones) in a swarm setting. To ensure the safety of their users, it is critical to verify the integrity of the software running on these vehicles. In the existing literature, the conventional software attestation developed for one node (vehicle) has been trivially extended for each node in a swarm. In this work, we explore the question of whether we could reduce the total computational cost of the software attestation of the swarm. To that end, we propose a novel program integrity verification mechanism for a single-node setting. The proposed mechanism is inherently designed considering the scalability in a swarm setting. Our initial results demonstrate that the proposed mechanism when utilized for a swarm significantly reduces the total swarm attestation time as compared to the prior art.**

Introduction: These days, vehicle nodes including automobiles and drones are being utilized in a *swarm* to handle a variety of critical tasks, such as geographical surveillance. Unfortunately, an attacker can compromise the effectiveness of such tasks by compromising only one of the nodes in the swarm. Hence, it is important to employ a *swarm attestation* mechanism that can be utilized for quickly validating the integrity of the homogeneous program applications running on all swarm nodes. We observe that a swarm attestation mechanism consists of two major sub-mechanisms: (1) individual program integrity verification (PIV) which checks the integrity of the software running on an individual node and (2) attestation topology which defines the sequence of the PIV executed on different swarm nodes. This implies that we can lower the total time for the swarm attestation by reducing the PIV time and selecting the appropriate attestation topology.

The existing swarm attestation schemes utilize different attestation topologies to cover all swarm nodes [2]. However, for PIV, they typically utilize the conventional software attestation (CSA) which validates the integrity by computing a cryptographic hash of the entire current static memory and comparing it with the pre-stored gold hash of the benign memory. Unfortunately, CSA hinders the performance of the application executing on the node. To balance the trade-off between the application execution time and the individual attestation time, PracAttest [1] divides the entire memory into multiple fragments, attests one fragment in every attestation instance, and facilitates application execution between attestation instances. While such a mechanism reduces the impact on the application execution, the total attestation time is much larger compared to the CSA scheme.

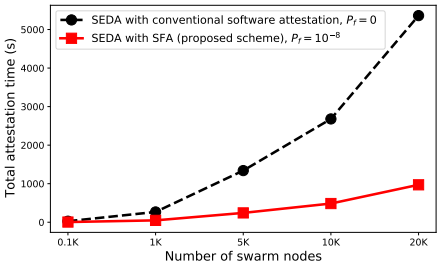

Fig. 1. Reduction in the swarm attestation time using the proposed scheme.

Contributions: In this work, we introduce Selective Fragment Attestation (*SFA*), a novel PIV scheme, which, in a single attestation instance, attests a large number of *randomly* selected small-sized fragments distributed across the program memory. The inherent motivation behind SFA is to probabilistically find at least a single fragment of memory that mismatches the pre-stored version and this happens quickly when the search gets distributed and many fragments (e.g., of size 8 bytes) are checked together in a single instance. PracAttest chooses a single large fragment (of size 4 kB) during an attestation instance which localizes rather than distributing the search. In a given attestation topology, SFA achieves a lower probability of failure (P_f) in detecting any modification in the program, performs the swarm attestation faster, and interrupts the application execution minimally compared to the prior art.

Testbed and Results: To evaluate the effectiveness of SFA, we first obtain the individual attestation time on a Raspberry Pi 3B board and then utilize the obtained time in OMNeT++ simulating the swarm setting discussed in SEDA [2]. In an experiment with a sample application (of size 8 MB) and malware (of size 40 kB), we observe that to achieve an average P_f of 10^{-8}, PracAttest needs to attest around 6 MB of memory leading to a mean attestation time of 54.6 seconds. For the same P_f, SFA needs to attest only 25 kB of memory consuming only 0.32 second. The degradation in the application execution time was also lower compared to PracAttest. Further, we also observe a significant reduction in the total swarm attestation time over the vanilla SEDA scheme as displayed in Figure 1.

References

[1] I. Abidi et al., "Practical attestation for edge devices running compute heavy machine learning applications," in *Annual Computer Security Applications Conference (ACSAC)*, 2021, pp. 323–336.
[2] N. Asokan et al., "SEDA: Scalable embedded device attestation," in *Computer and Communications Security (CCS)*, 2015, pp. 964-975.

Symposium on Vehicles Security and Privacy (VehicleSec) 2023
27 February 2023, San Diego, CA, USA
ISBN 1-891562-88-6
https://dx.doi.org/10.14722/vehiclesec.2023.23089
www.ndss-symposium.org