Secure and Efficient Monitoring of Confidential VMs using eBPF

Kanta Uesugi
Kyushu Institute of Technology
uesugi@ksl.ci.kyutech.ac.jp

Kenichi Kourai
Kyushu Institute of Technology
kourai@csn.kyutech.ac.jp

1 Background

In clouds, there is a risk of insiders eavesdropping on sensitive information in users' virtual machines (VMs). To counteract this risk, recent clouds provide confidential VMs, which are protected from insiders by trusted execution environments (TEE) such as AMD SEV. Since confidential VMs cannot prevent intrusion into VMs, intrusion detection systems (IDS) are still required. However, IDS cannot monitor confidential VMs from the outside using VM introspection (VMI) because the memory of VMs is protected.

SEVmonitor [1] enables IDS to monitor confidential VMs by communicating with an agent securely running inside the VMs. IDS running in a dedicated confidential VM can securely obtain requested memory data and analyze kernel data using VMI. The drawback of SEVmonitor is that monitoring performance is largely degraded due to the communication overhead. Particularly, when IDS traverses kernel data structures with pointers, it needs communication to obtain memory data one by one. According to our experiment, it took 27x longer to obtain information on all processes, compared with traditional VMI.

2 eBPFmonitor

We propose eBPFmonitor, which enables IDS to efficiently monitor confidential VMs by reading ahead kernel data using eBPF inside VMs. As shown in Figure 1, eBPFmonitor isolates the target system using a container created in a confidential VM and securely runs an agent outside it. When IDS needs to access kernel data structures with pointers, it injects custom eBPF programs into the kernel of the target VM via the agent. For example, an injected eBPF program traverses the list of processes, obtains necessary memory data, and returns it to the IDS in a batch. The injected eBPF programs can be safely executed because the verifier can detect illegal instructions and infinite loops. As such, eBPFmonitor can reduce communication overhead between IDS and the agent and improve monitoring performance.

eBPFmonitor loads eBPF programs into the target VM in advance when IDS starts monitoring the VM. When the IDS needs specific kernel data, the IDS sends a request to the agent. The agent executes one of the loaded eBPF programs, whereas the eBPF program collects the addresses of the memory where the kernel data is stored. In parallel, the

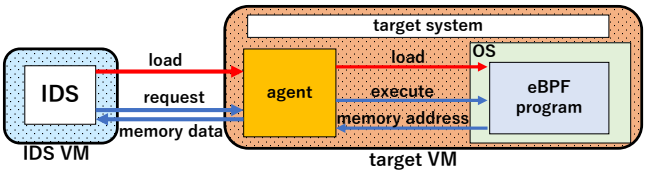

Figure 1: The system architecture of eBPFmonitor

agent obtains the memory data corresponding to the collected addresses from the kernel and sends it back to the IDS sequentially. To protect the memory data, the kernel encrypts it so that only the IDS can decrypt it. The IDS stores the received memory data in the cache and uses it when necessary.

We have implemented eBPFmonitor in Linux and KVM using BPF CO-RE. BPF CO-RE enables developed eBPF programs to run in different kernel versions. Specifically, BPF CO-RE compiles eBPF programs including type information and adjusts them to the kernel version of the target VM at load time. However, BPF CO-RE cannot handle kernel global variables, e.g., the list head of the kernel modules. eBPFmonitor makes global variables available by converting them from symbol names to kernel addresses at runtime.

We conducted experiments to show the effectiveness of eBPFmonitor. We measured the time taken to traverse the list of processes, the list of kernel modules, and the hash table of TCP sockets in a confidential VM. For comparison, we measured the time in SEVmonitor, which communicated between IDS and the agent whenever the IDS needed kernel data. Also, we applied traditional VMI to a non-confidential VM. As a result, eBPFmonitor was 43% and 36% faster than SEVmonitor for processes and kernel modules, respectively. However, it was 9.6% slower for TCP sockets due to the overhead of the eBPF program. Compared with VMI, it was still 4.6-15.4x slower.

References

[1] T. Nono and K. Kourai. 2022. Secure Monitoring of Virtual Machines Protected by AMD SEV in Public Clouds. *SAES 2022*.