Introduction

- At times, user does not trust the host running the application.OS can act malicious and application need to prevent its data from the privileged code.
- The paper proposed a mechanism which uses secured and trusted hardware for protecting user-level architecture from privileged system software.

Desirables from SCONE

- Small TCB => less attack surface => easier to protect application
- Low overheads
- Transparent to docker

Secure Containers

Namespace Isolation:

- In containers, processes are launched in different environment isolated by namespaces
- VMs use tighter isolation than the containers
- Containers share the same kernel and isolation is dependent on host kernel's capbilities.
- VM have fine control over resources while container uses cgroups for limiting, isolating and accounting of resources.

Intel SGX - Enclave life cycle

- ECREATE -> EADD -> ENIT -> EENTER -> EXIT
- Entering in and out of enclave is costly

Design TradeOffs

External Interface

- The author uses an intermediate design for isolating interface of trusted and untrusted code.
- Target is to reduce TCB and have less exits from enclave

System Call overhead

- Copying syscall arguments from enclave memory to non-enclave memory
- Entering and exiting out of the enclave

Memory Access Overhead

- MEE encrypts and decrypts data while fetching it from DRAM to cacheline
- Data in cacheline is in plaintext. It is difficult to prove CPU traffic than memory traffic which is organised.
- When application size is beyond EPC,eviction cost

Architecture

M:N threading

- scheduler schedules m application threads on n OS threads
- No preemption required as application code is trusted by application scheduler and preemption will complicate the design
- Reduced exits from enclave as scheduler is inside the enclave

Why threads go for exponential backoff?
Assuming past=future, if the thread has waited for 1 seconds and nothing has happened.So, it will assume that nothing will happen for another 1s and it will wait for 2s.

Asynchronous Syscalls

- Separate syscall threads for executing syscalls
- Checks on the pointers passed by kernel (similar checks are done by kernel for user pointers)
- IAGO attacks - check kernel do not pass user space pointer (similar checks are done by kernel for kernel spacer pointers)

Shielding Layers

- Transparently encrypts and decrypts data
- Prevents malicious pointers

Replay Attack The traffic in past is repeated again.SCONE uses identifier to prevent this.

Where keys are stored?

- Key and certificate for network encryption is present in filesystem
- The keys with which filesystem is encrypted is presented in FS protection file in image
- FS protection file is again encrypted with the key but it is not present in the image

- Final key is passed to the container only after it is ensured that container is secured
- The symmetric keys for encryption of console stream are passed at runtime by scone client.

How image is trusted?

- Image is build in a secure environment by the container owner or trusted party
- Final key is passed to the container only after remote cryptographic attestment of container with the help of hardware

Ephermal FS

- Implementation by SCONE for a file system present in memory only.
- Each time, container is started again FS rollbacks to initial stage.

Discussion on Section 4

- In redis benchmarks, scone performs poorly due to lack of parallelism
- In Memcached benchmarks, scone performs better due to faster implementation of network shield
    - Graph have a turn due to livelock like situation
    - With increased input throughput, scheduler takes poor decision and net output throughput is decreased.
- In NGINX and apache , scone performs poorly than glibc application
- In general, scone async performs better than scone sync

Syscall Benchmarks

- In smaller buffer case
    - Scone async performs better than scone sync due to reduced exits from enclave
    - Number of syscall increase for both native and scone-sync due to increased system calls and saturates after some time due to kernel-level contention
- In Large buffer cases
    - Difference between scone-async and scone-sync reduced as overheads for copying buffers from enclave memory to DRAM are the bottleneck.