Confuzer: System Call Fuzzer for Understanding Secure Container Mechanism

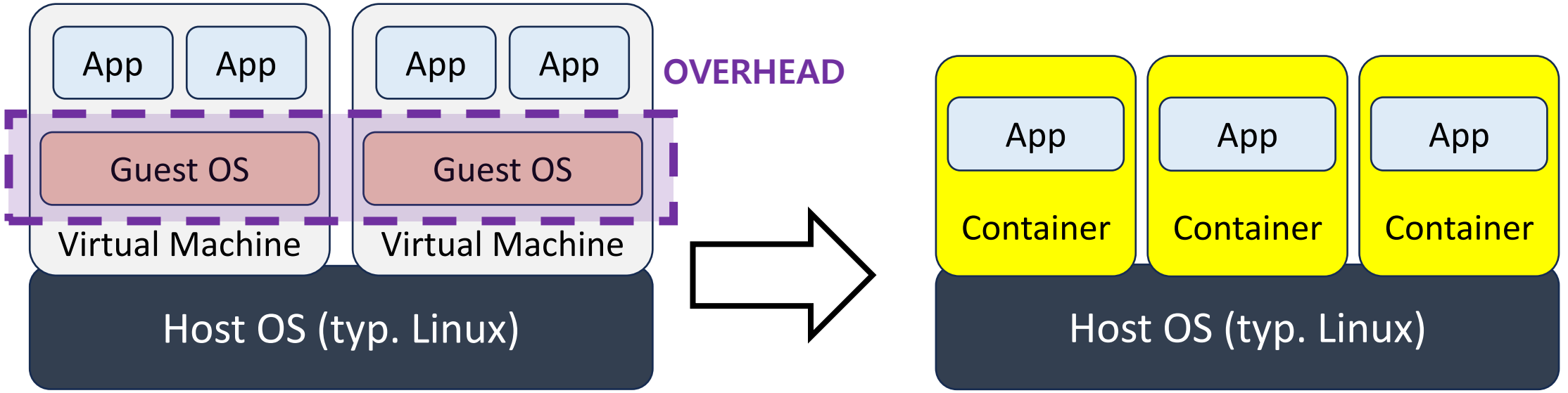
Hyeonseok Shin, Hosang Yoo, Yongwon Lee, Minjung Jo

(Advisor: Byungchul Tak)

Kyungpook National University (KNU), Daegu, Republic of Korea

Container

In the past, cloud environments have used VM technology,
but nowadays, mainly use Container technology

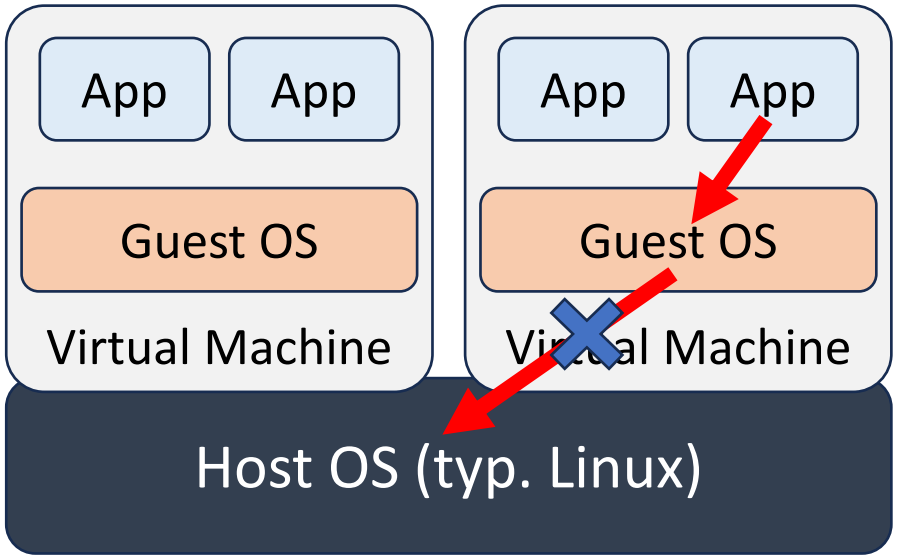


VMs are Heavy because they contain guest OS

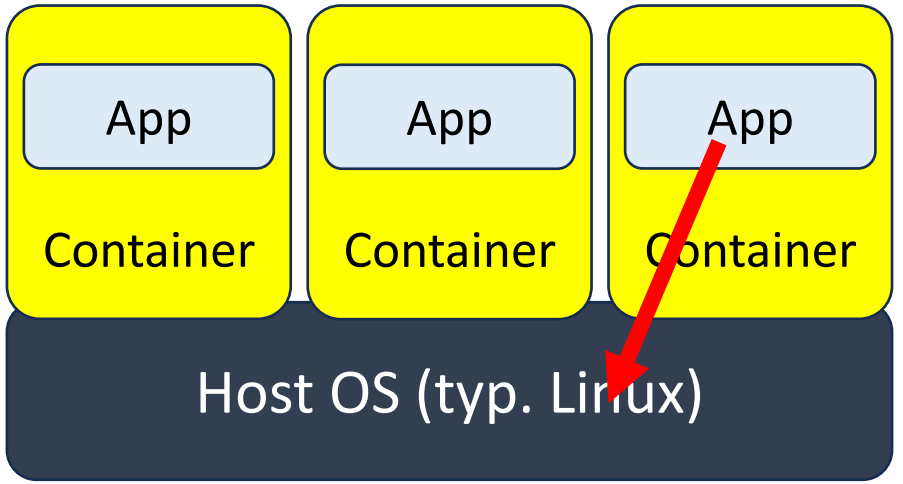
Containers only share a single Host OS

Container Security Problem

Because Container shares the host OS, it is possible to access the host kernel and affect the entire system through Privilege Escalation by invoking dangerous system calls

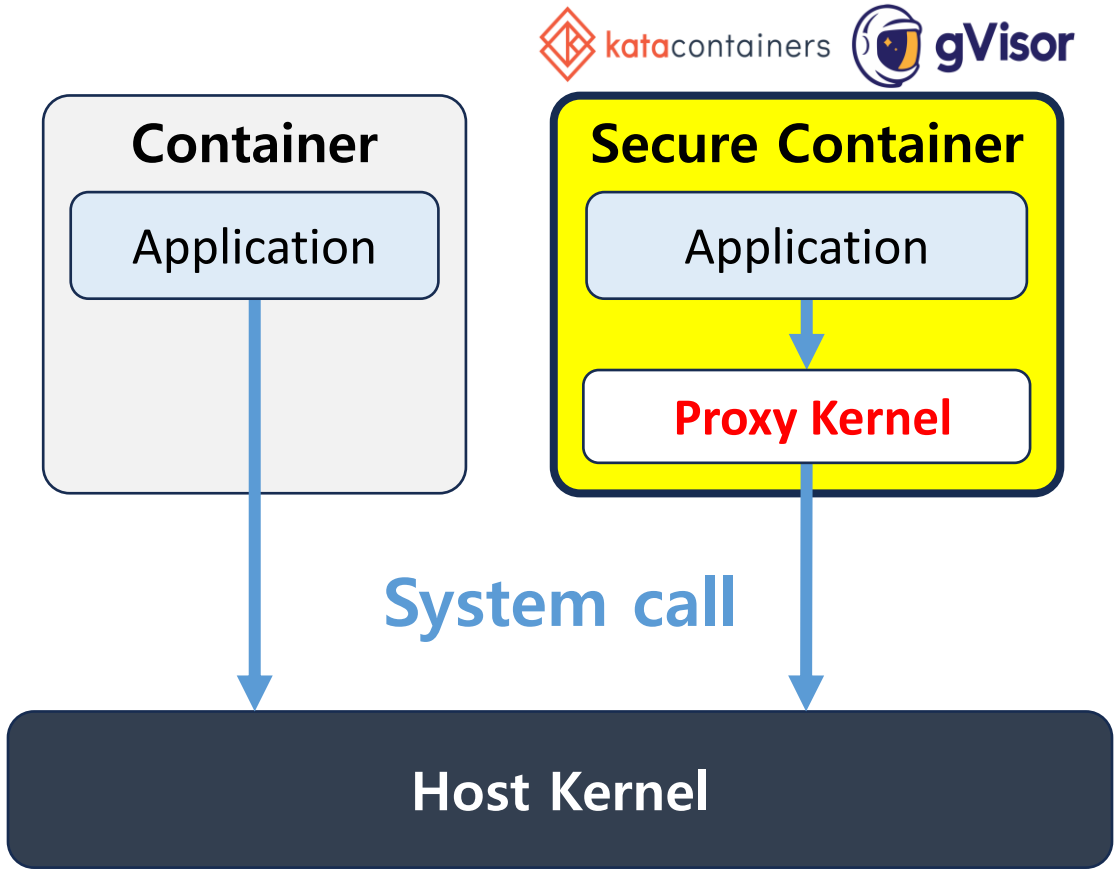


VM's application can't access to Host OS



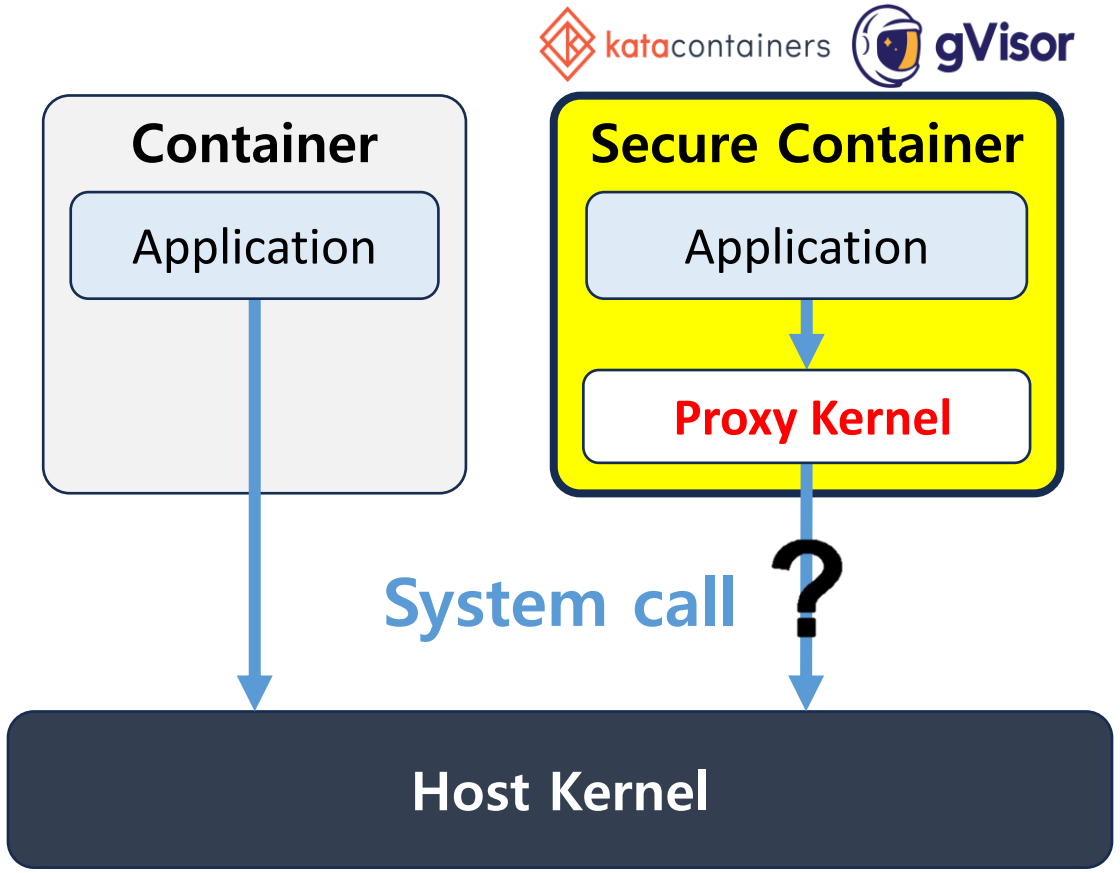
Container's application can access to Host OS

What is Secure Container?



- **Secure containers add a proxy kernel to prevent the container from directly accessing the host kernel**
- **Currently, this approach has led to various secure containers, such as gVisor and Kata Containers**
- **The Proxy Kernel is unable to determine which system calls are being invoked**

What is Secure Container?



- **Secure containers** add a proxy kernel to prevent the container from directly accessing the host kernel
- Currently, this approach has led to various secure containers, such as **gVisor** and **Kata** Containers
- The **Proxy Kernel** is unable to determine which system calls are being invoked

Goals

Goal 1. Fuzzing on *All* System Call

Utilize ***Confuzer*** to fuzz all system calls for all possible arguments

Goal 2. Expand *Observability* for Secure Container Mechanism

Analyze output system calls and ***understand the secure container's mechanism***

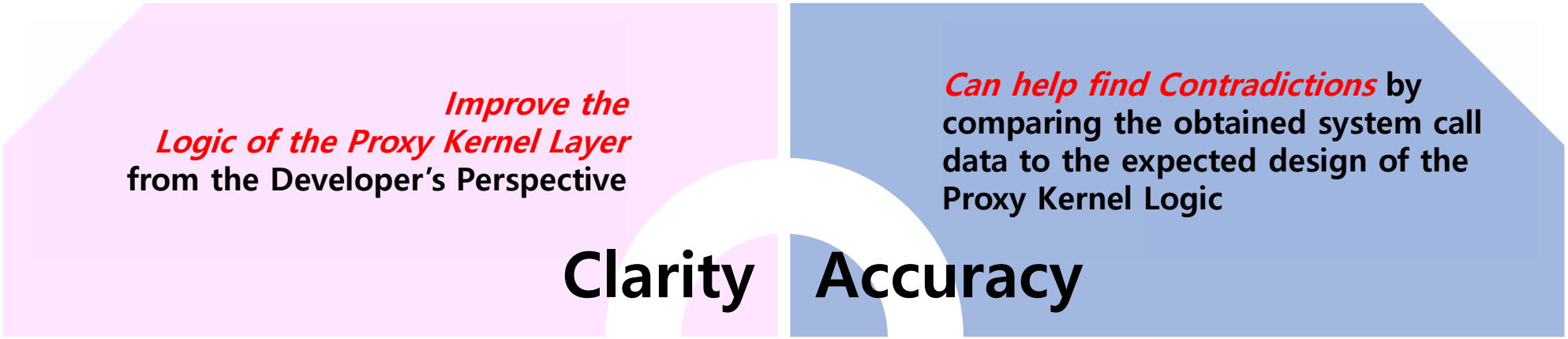
Advantages of Research

Advantages of Research

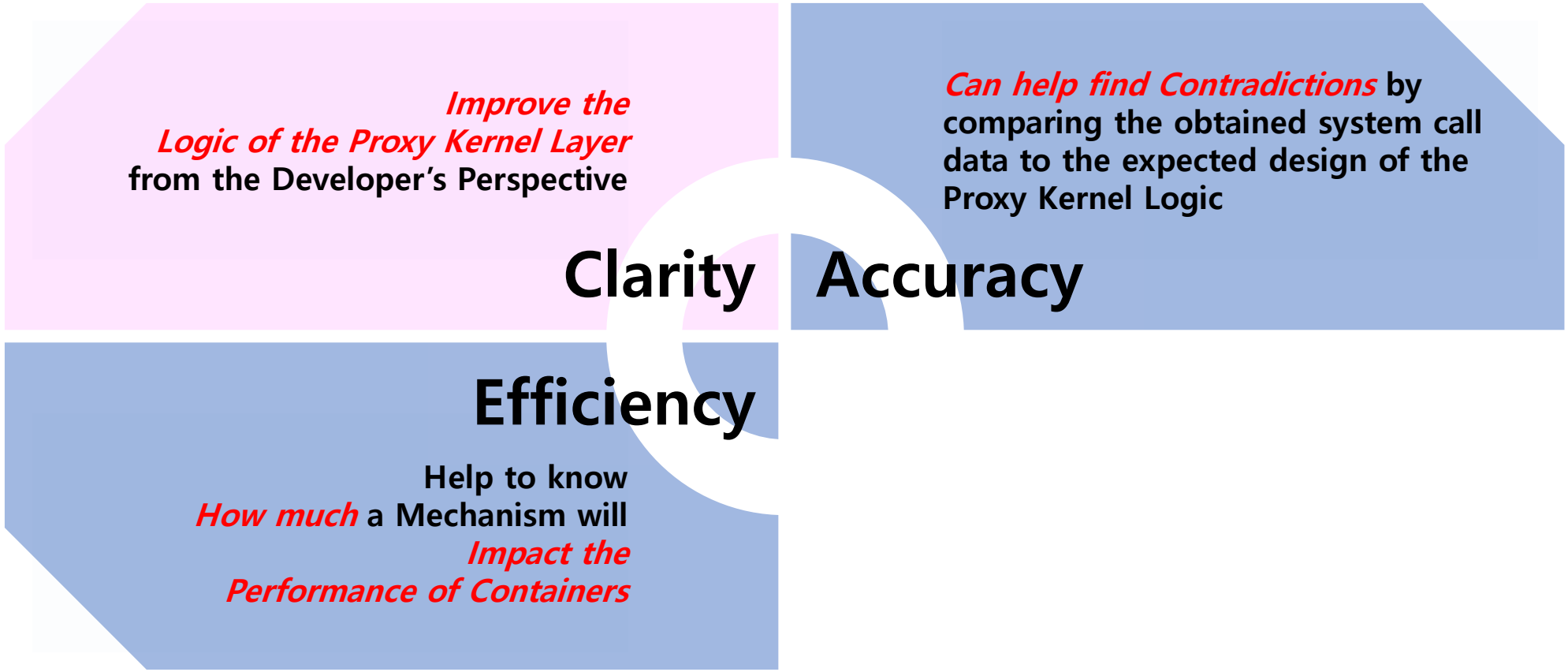
Can help find Contradictions by comparing the obtained system call data to the expected design of the Proxy Kernel Logic

Accuracy

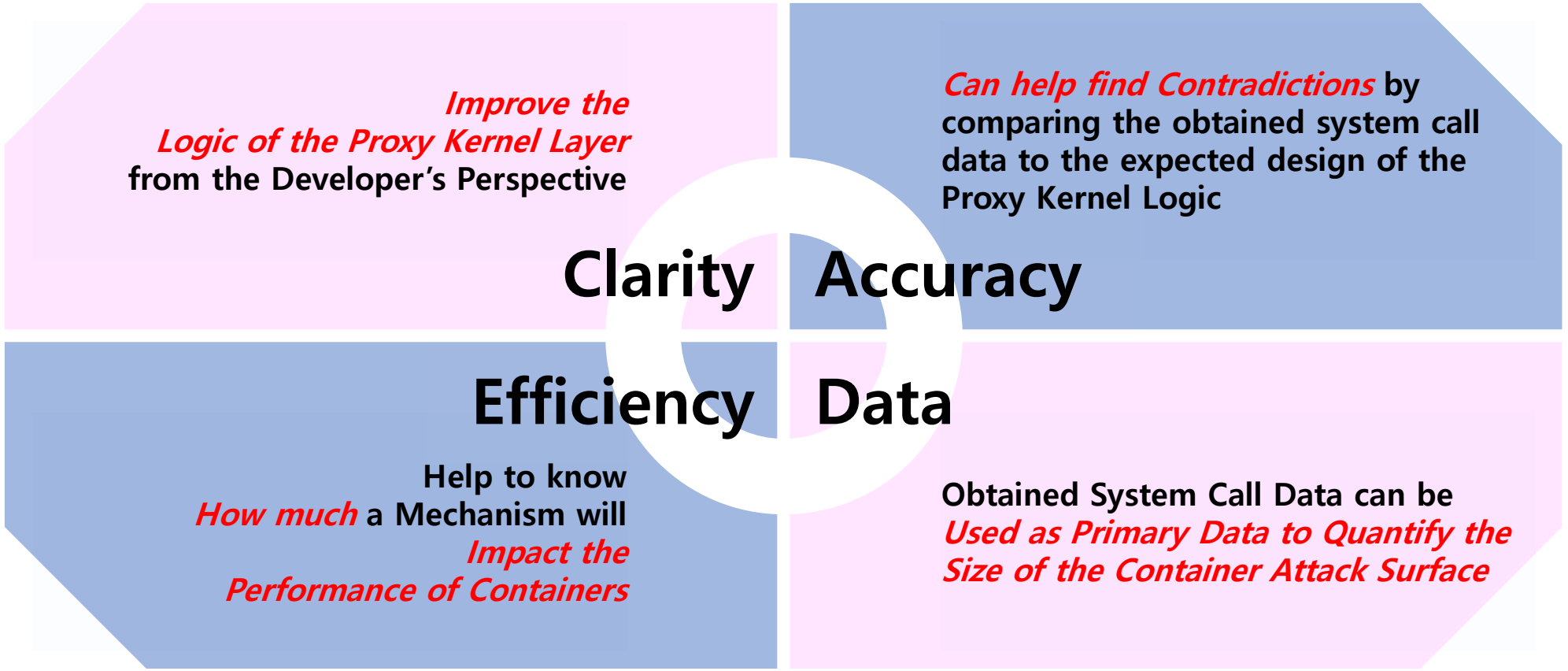
Advantages of Research



Advantages of Research



Advantages of Research



Challenges

Of the many system calls observed in the Host Kernel,
it should be possible to
Extract Only Certain Test System Calls

Open(**path A**, arg2, arg3)

≠

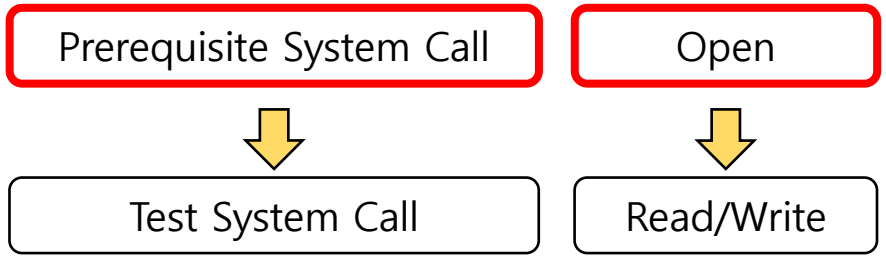
Open(**path B**, arg2, arg3)

Some arguments affect observed system calls,
so **Tests Should be Conducted with All Possible Arguments**

Ex) Open system call produces ***Very Different Results***
when inserting a path inside a container than
when inserting a path to storage shared with the host

Since there are ***System Calls that Require Preparation***
rather than system calls that can be executed alone,
the ***Preparation Process Must be Resolved***

Ex) To test the *read/write* system call,
open must precede,
and this preparation process must be performed normally



Challenges

Of the many system calls observed in the Host Kernel,
it should be possible to
Extract Only Certain Test System Calls

Open(**path A**, arg2, arg3)

≠

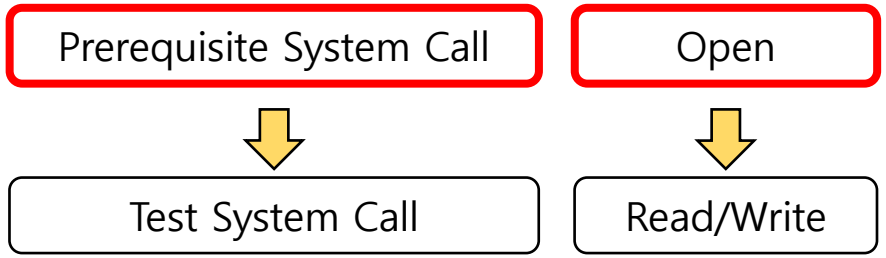
Open(**path B**, arg2, arg3)

Some arguments affect observed system calls,
so **Tests Should be Conducted with All Possible Arguments**

Ex) Open system call produces ***Very Different Results***
when inserting a path inside a container than
when inserting a path to storage shared with the host

Since there are ***System Calls that Require Preparation***
rather than system calls that can be executed alone,
the ***Preparation Process Must be Resolved***

Ex) To test the *read/write* system call,
open must precede,
and this preparation process must be performed normally



Challenges

Of the many system calls observed in the Host Kernel,
it should be possible to
Extract Only Certain Test System Calls

Open(**path A**, arg2, arg3)

≠

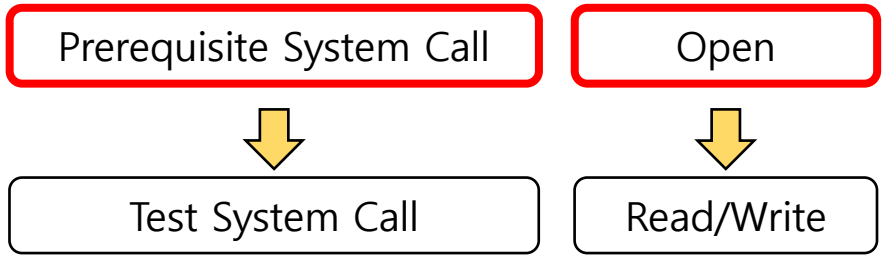
Open(**path B**, arg2, arg3)

Some arguments affect observed system calls,
so **Tests Should be Conducted with All Possible Arguments**

Ex) Open system call produces ***Very Different Results***
when inserting a path inside a container than
when inserting a path to storage shared with the host

Since there are ***System Calls that Require Preparation***
rather than system calls that can be executed alone,
the ***Preparation Process Must be Resolved***

Ex) To test the *read/write* system call,
open must precede,
and this preparation process must be performed normally



Challenges

Of the many system calls observed in the Host Kernel,
it should be possible to
Extract Only Certain Test System Calls

Open(**path A**, arg2, arg3)

≠

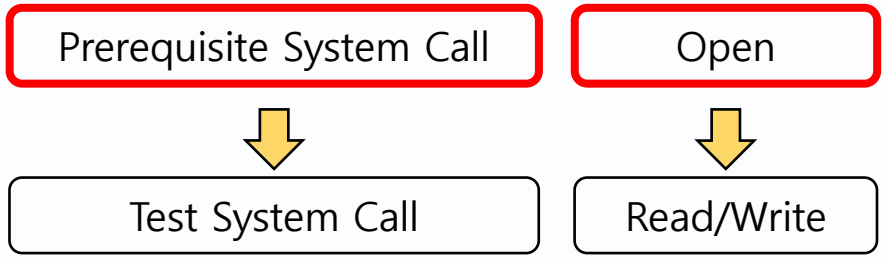
Open(**path B**, arg2, arg3)

Some arguments affect observed system calls,
so **Tests Should be Conducted with All Possible Arguments**

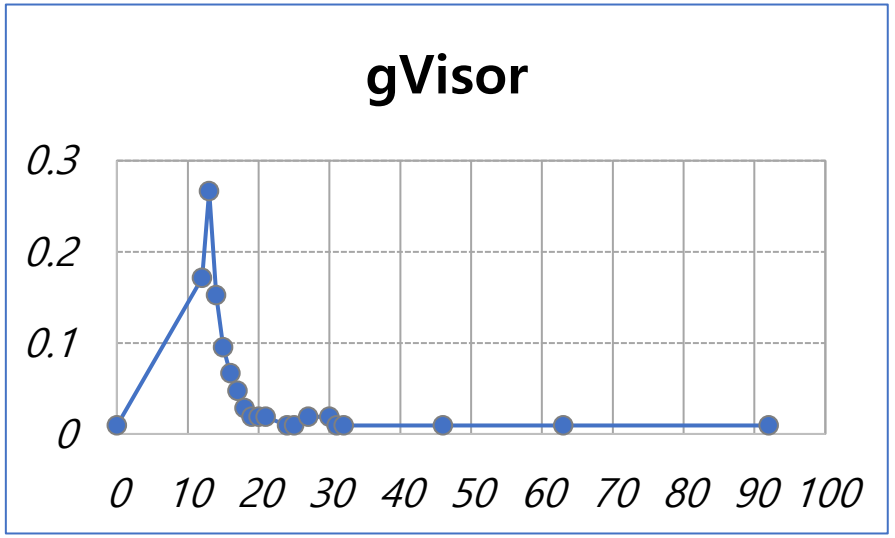
Ex) Open system call produces ***Very Different Results***
when inserting a path inside a container than
when inserting a path to storage shared with the host

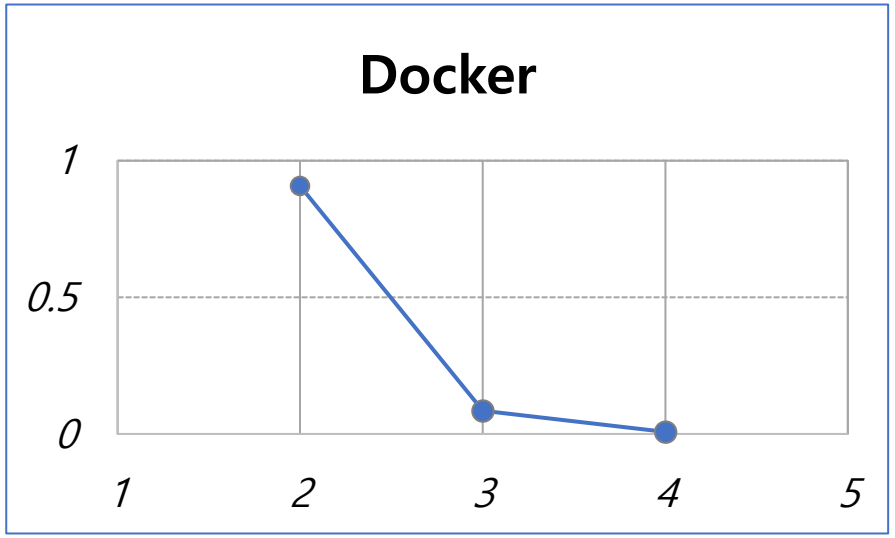
Since there are ***System Calls that Require Preparation***
rather than system calls that can be executed alone,
the ***Preparation Process Must be Resolved***

Ex) To test the *read/write* system call,
open must precede,
and this preparation process must be performed normally



Current Results and Findings



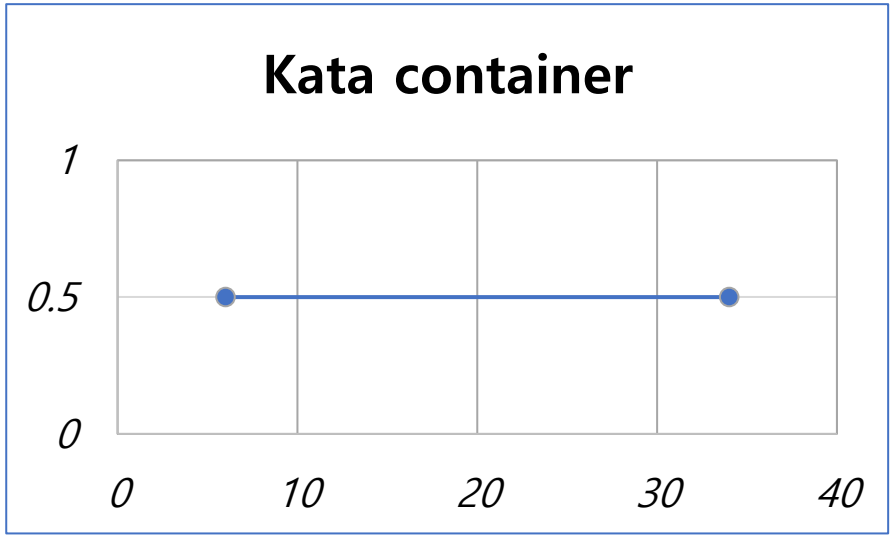


- **X-axis**

The number of system calls generated by the runtime as a response to single system call

- **Y-axis**

The proportion of system calls



The End