WP - 2

Framework for evaluating the cost of security of IoT processors

PRESENTED BY ISMI ABIDI

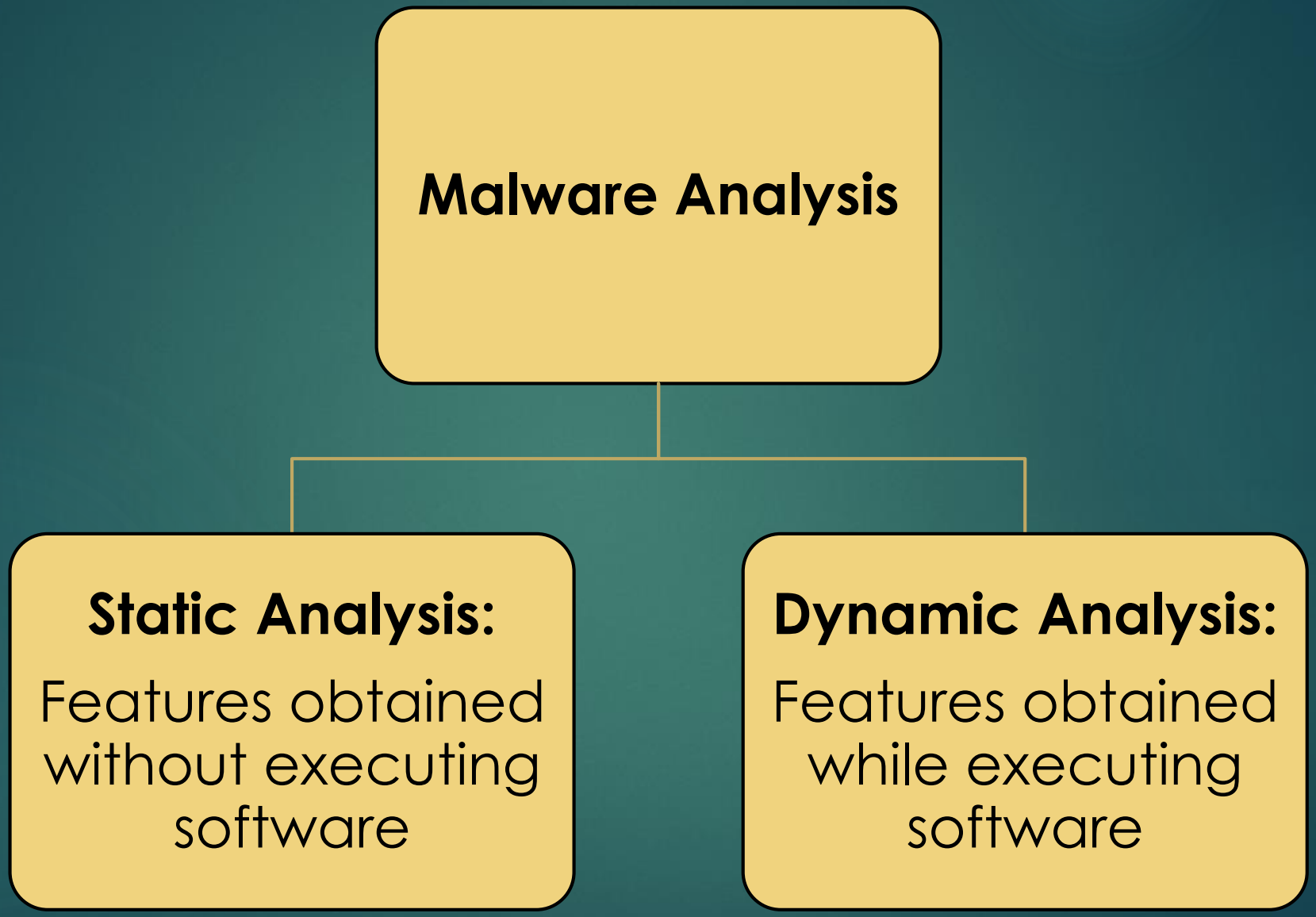
IIT DELHI

Contents

- Motivation
- Proposed solution
- Framework to measure perfrormance
- Future Work
- Demo

Motivation

- Increase in IoT devices lead to increase in Malware Attacks
- PhishMe has released its latest malware trends report for Q1 2017, concluding that botnet activities has risen by 69%.
- What is Malware
  - Intrusive software includes worms, trojans, botnets, ransomware, adware, spyware


Malware Analysis
Static Analysis:
Features obtained without executing software
Dynamic Analysis:
Features obtained while executing software

Static Features

- control flow graph
- Function call graph
- API call and opcode sequences

Dynamic Features

- Instruction traces
- Registry changes
- Memory writes

Dynamic Features more useful in current scenario

Malware Detection

Signature Based

- Repository of known malware
- Examine code to match patterns
- Ineffective for new or variants of existing malware

Behavior Based

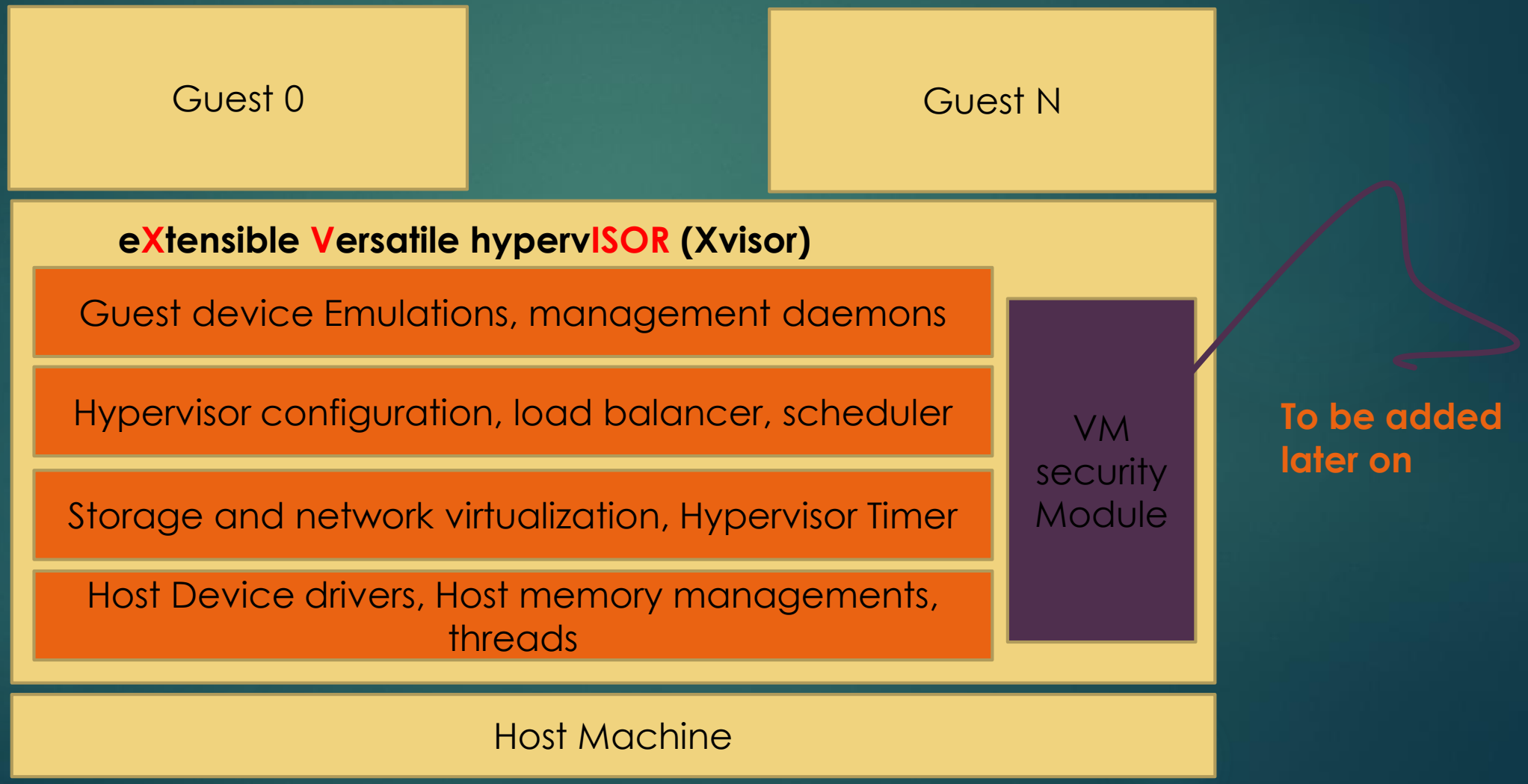
- Actions performed during execution
- Examine behavior instead of code

	Signature Based	Behavior Based
Pros	• Effective against known malwares • Fast • Widely used	• Works well for unknown malwares • Identifies what the malware does in a specific environment
Cons	• Can't detect unknown malwares (polymorphic or metamorphic or zero day attacks) • Database Management	• High false positives • Heavy computation

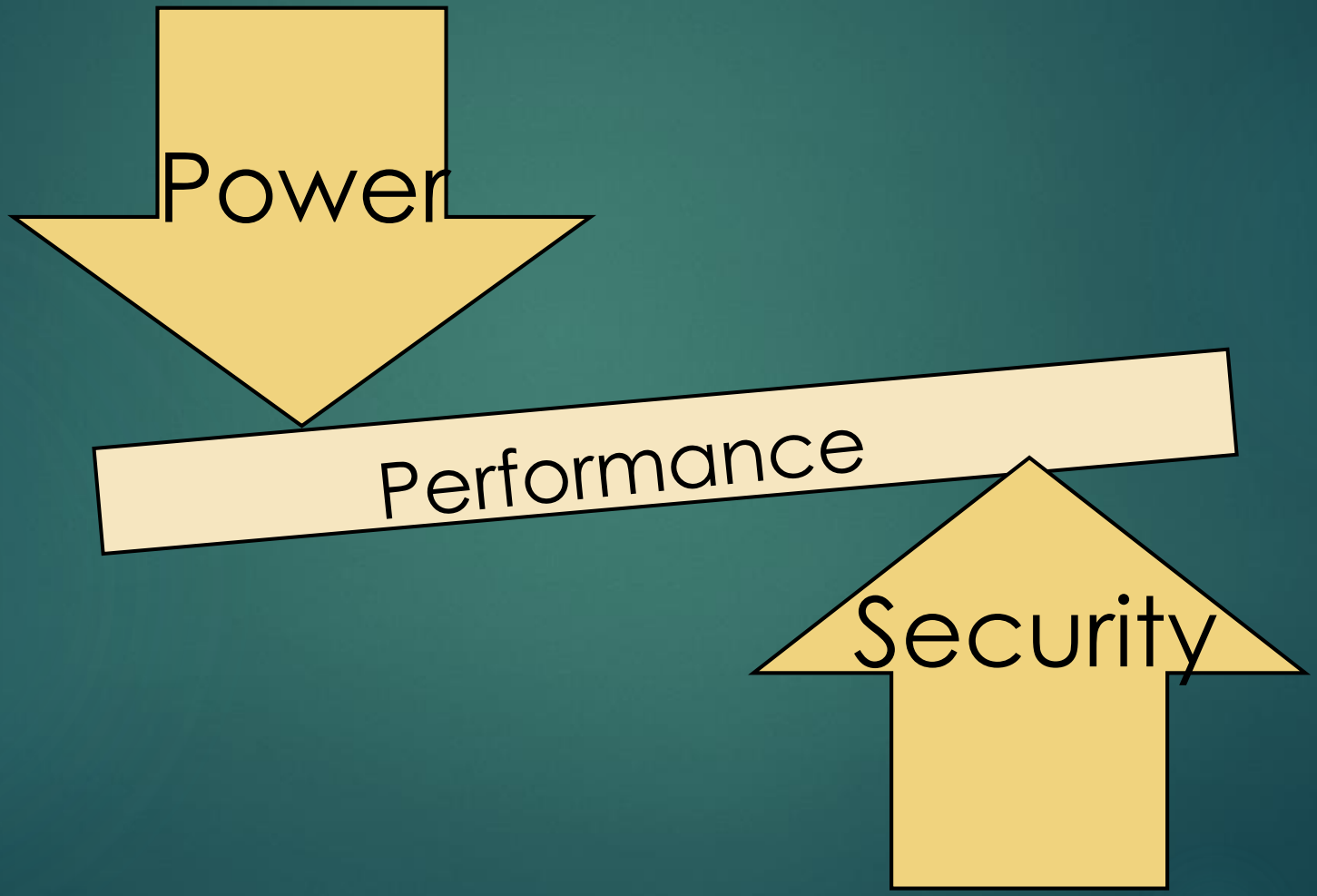
XVISOR

- Hardware assisted system virtualization software
  - Type 1 hypervisor
- High performance and low memory footprint
  - ARMv5, ARMv6, ARMv7, ARMv8a, x86_64
- Can run on device (ARM boards) as well as Qemu

System Virtual Machine Model implemented by Xvisor



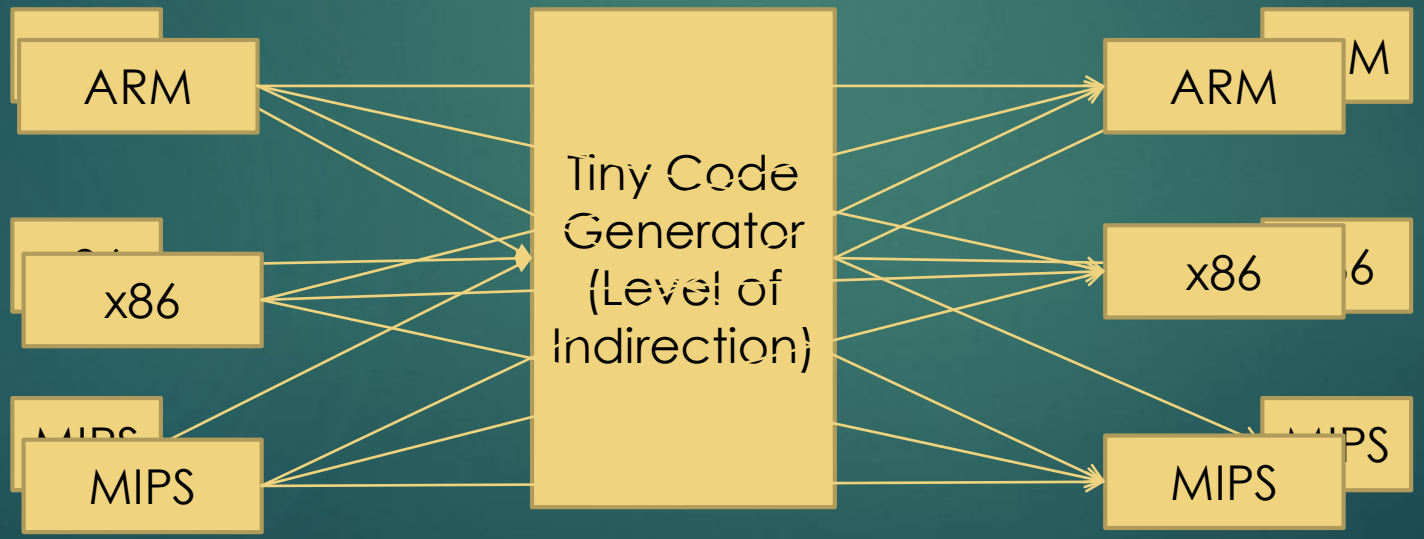
Goals/ Aim



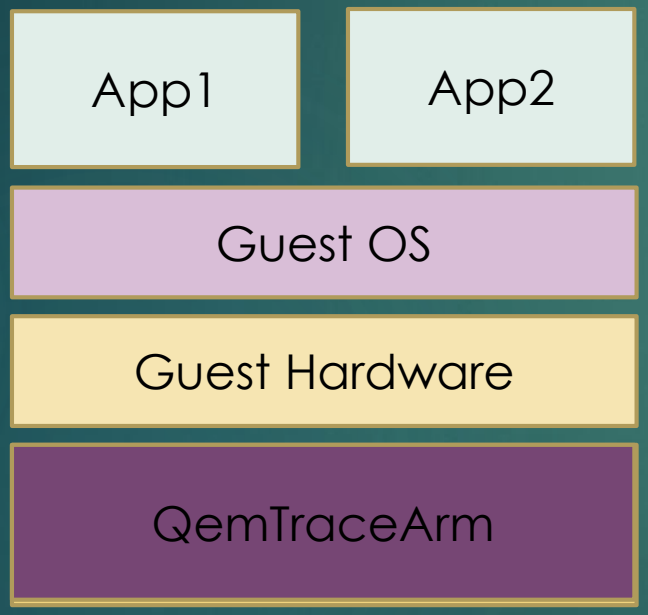
QEMU- Quick Emulator

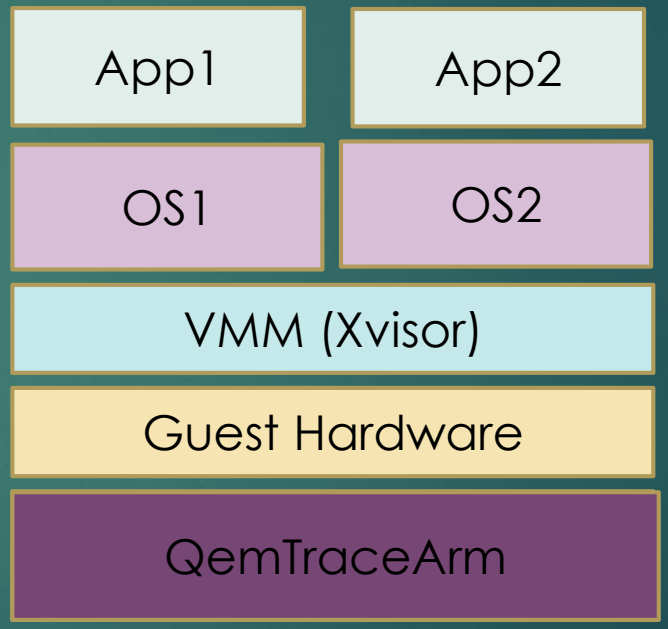
- Machine Emulation and Virtualization
- Modes
  - User Mode
    - Linux process built for one CPU executes on another CPU
  - System Mode
    - allows emulation of a full system, including processor and assorted peripherals

Dynamic Binary Translation



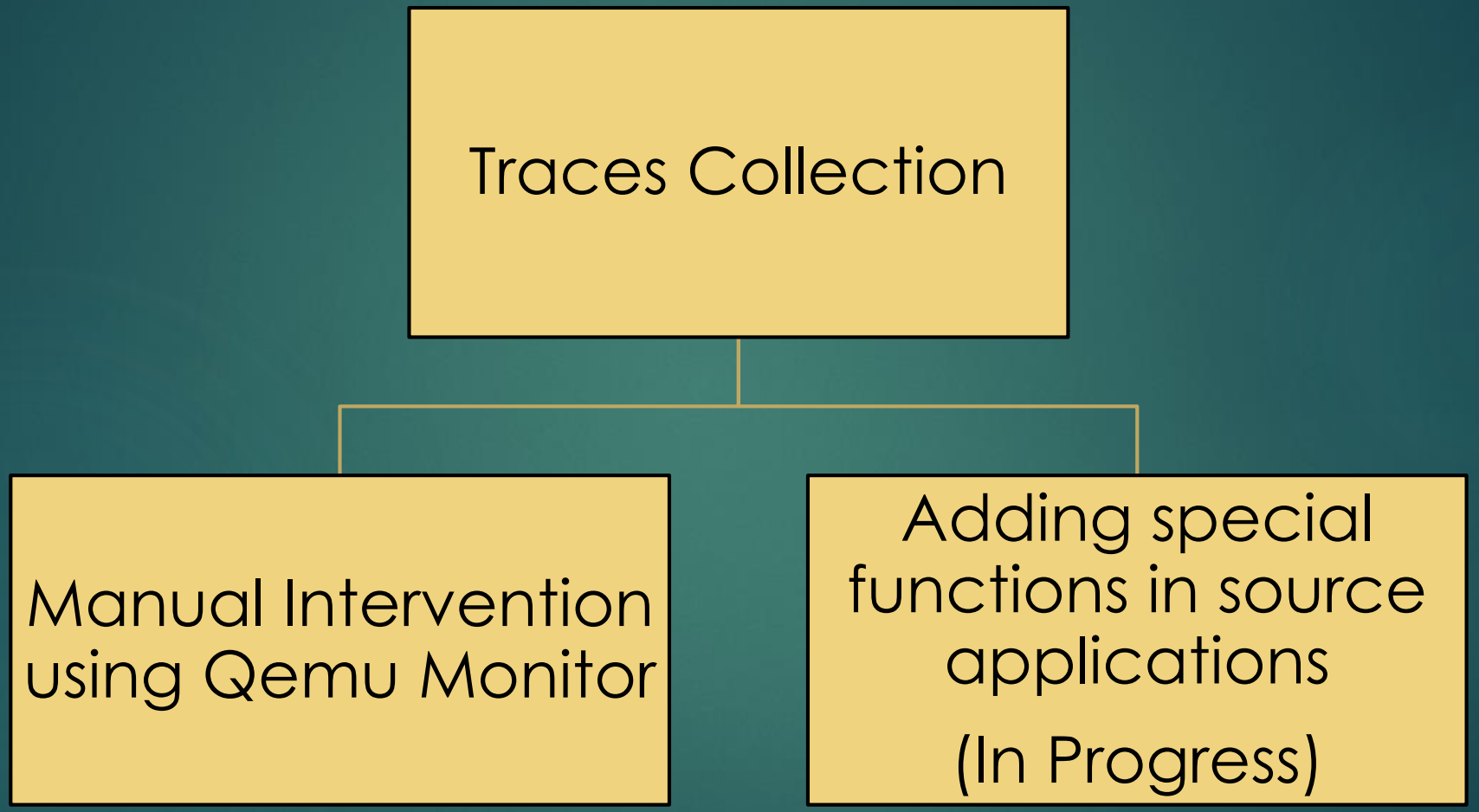
Qemu for system mode





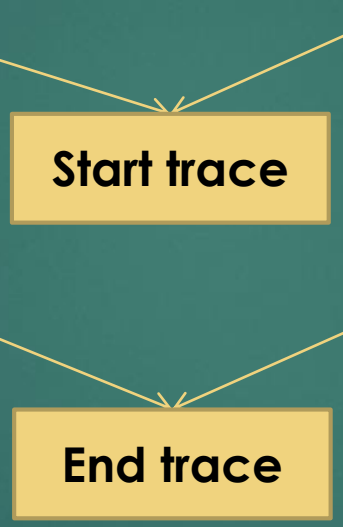
Collecting Trace Using QemuTraceArm

- Boot and Login into guest OS
- Initialization
- Execution Traces
- Cleanup
- Free memory and Power off system


Traces Collection
Manual Intervention using Qemu Monitor
Adding special functions in source applications
(In Progress)

Adding special Instructions

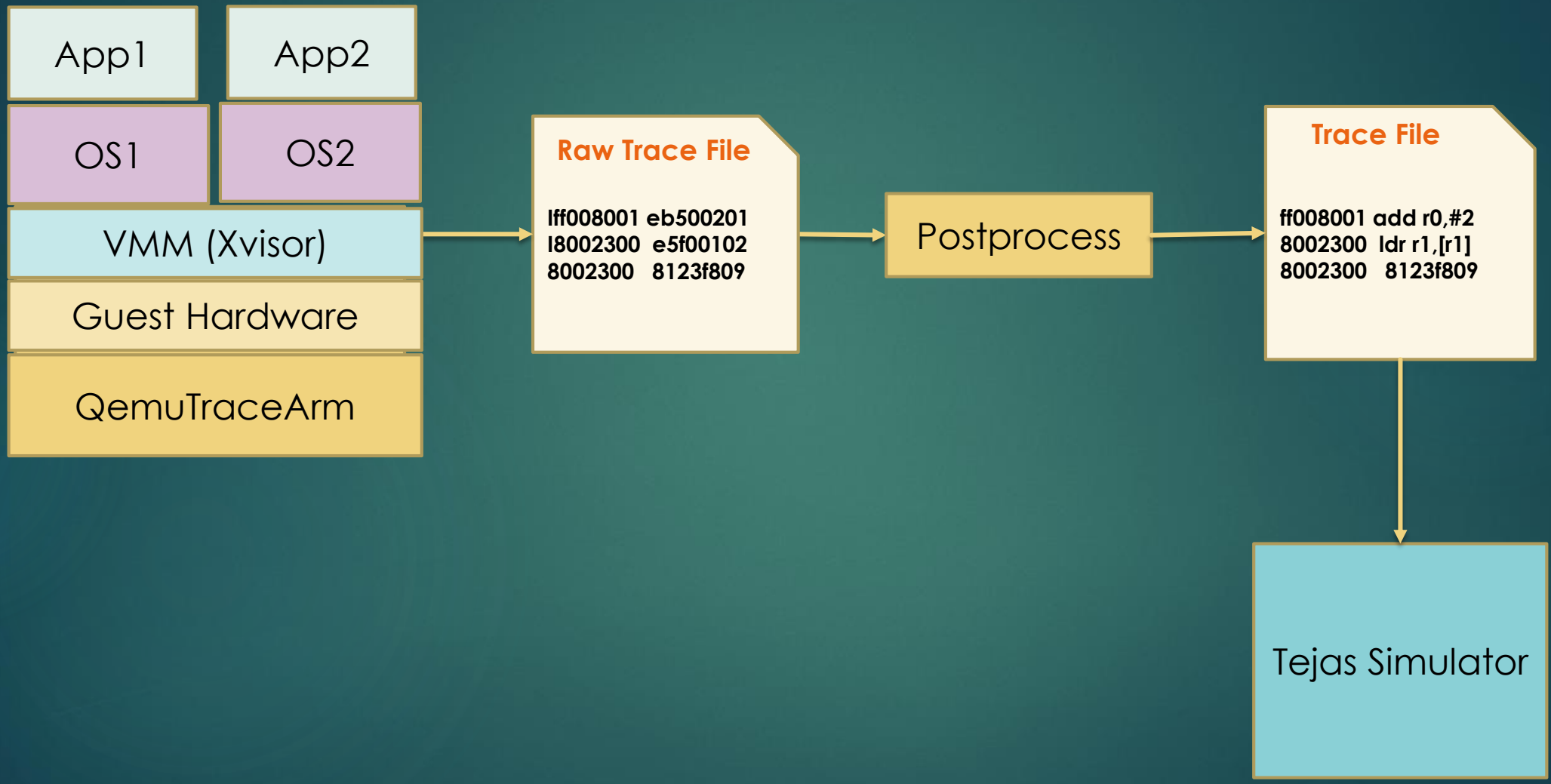
- Add instructions
  - asm (“int $100”);
  - asm (“int $101”);
- Qemu monitor unavailable



Manual Intervention

- Commands on qemu monitor
  - qqq_start_trace
  - qqq_end_trace
- Propietry Sofware

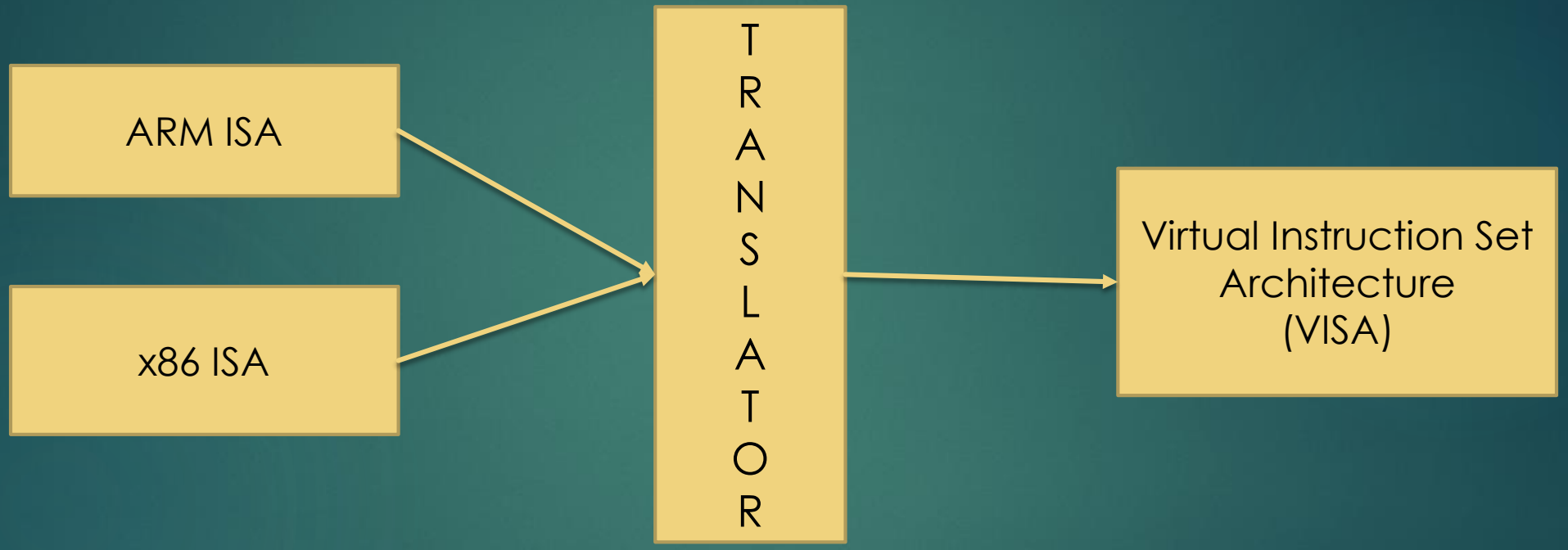
Running Traces on Tejas

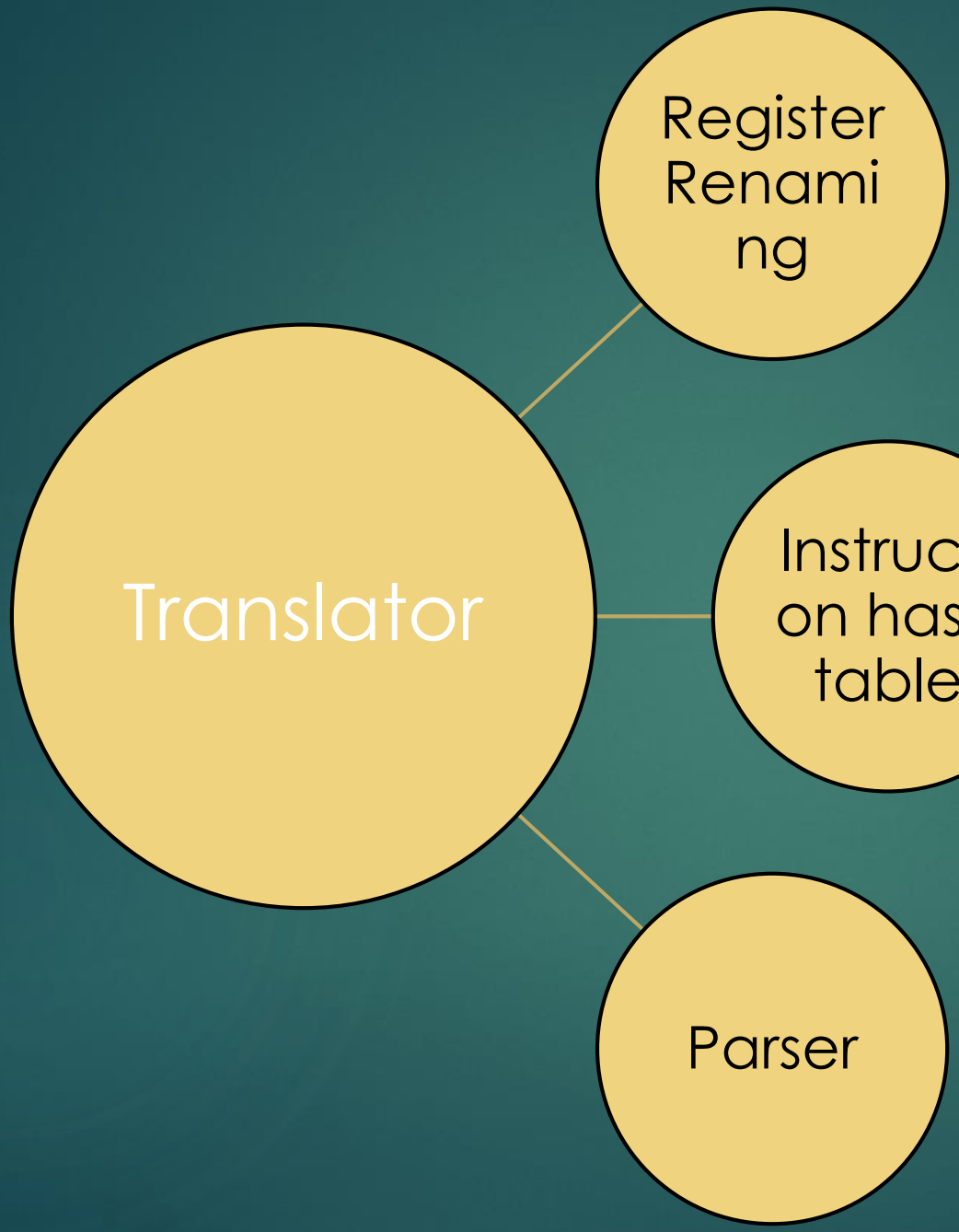


TEJAS- The Efficient Java Simulator

- In-house multicore architectural simulator
- Simulates x86 ISA
- Fast
- Output
  - Power based on Mcpat
  - Performance based on L1 cache hit, L2 cache hit, IPC

Translator


ARM ISA
x86 ISA
TRANSLATOR
Virtual Instruction Set Architecture (VISA)



- Map registers of source ISA to target ISA
- Add temporary registers

- Create handle for each source instruction

- Define call back in target architecture for every instruction in source architecture

Future Work

- Analyse Power-Security-Performance trade-off
- Malware detection
  - Here, it is possible to make new algorithms that are more power efficient (mainly because we are targeting very power constrained devices)
  - Example problems: power efficient HMMs (hidden Markov Models), Bayesian analyzers, and anomaly detection
  - Reduce the false positive rate (collaborative filtering of malwares)
- VM Checkpointing based strategies for recovery

THANK YOU