School of Engineering & Applied Science
THE GEORGE WASHINGTON UNIVERSITY

Rethinking Embedded System Virtualization

Robert Gifford, Lucas Baier, Gabriel Parmer
{robertgif, lbaier, gparmer}@gwu.edu
Computer Science, The George Washington University

GW

Motivation

Embedded Systems (ES)

- Computers that interface with the real world

- Found in day to day life
- 1 car ≈ 60 ES

Problem: Security

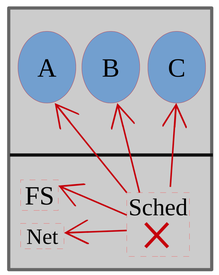
- ES now connecting to "The Cloud"
- Internet = More Vulnerabilities
- Need for improved systems + security

Solutions:

- Virtualization
  - The path to isolation
  - Isolation = increased fault tolerance
- Smaller System Components:
  - Greater compositional flexibility
  - A sleeker, more tailored lightweight system
- Use of Legacy code:
  - No longer need heavy man power
  - Tested code = safer code

Current Virtualization

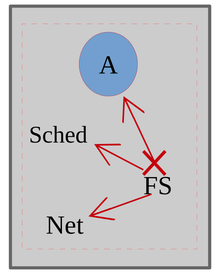
No Virtualization



Linux

+ lightweight
- No isolation
- No legacy support
- low fault tolerance

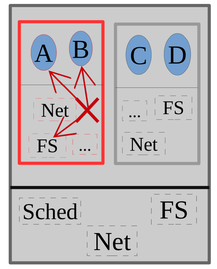
Unikernel



MirageOS

+ Some legacy support
- Supports only one application
- No isolation

Full VM



Xen

- Heavyweight
- Memory intensive
- Expensive I/O Path

A *New* Embedded System Virtualization

The RUMP Kernel + eVM

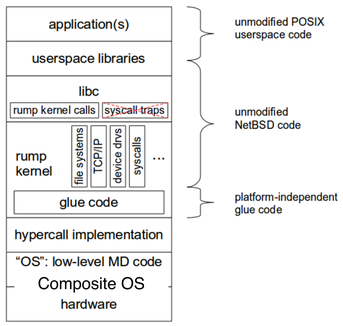


Image from Kantee, Antti. "The Design and Implementation of the Anykernel and Rump Kernels"

+ Lightweight
+ Legacy support
+ Isolation
+ Scalable kernel size
+ Multiple Applications

Introducing the Embedded Virtual Machine (eVM)

-Uses the Composite OS (COS) as host

-Leverage COS's philosophy of fine grained components

-Runs RK as an isolated user space component

-Ability to pick and choose system services per RK

-COS exports a layer of "hypercalls" to the RK

-Creates an app specific environment

-All possible thanks to COS's fast InterProcess Communication (IPC)

The Rump Kernel (RK)

Insert Rump Pic

+Out of the box NetBSD code
+Tried and tested Drivers
+Scalable Kernel Size

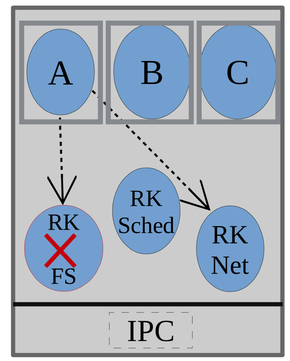
Results

Our new virtualization platform (eVM) allows for:

Porting of legacy code +
Application environment isolation +
Carve out unneeded code +
COS hypercall layer
=
a sleeker, more tailored system +
a safer system +
faster system development

Future Work

Keep Pushing Granularity



Composite + RK

- Application specific protection domains
- RK component specific protection domains
- Treat each ported RK component as a server in user space