Virtualization

- Guest OS expects to access hardware directly.
- Most importantly, expects to operate in supervisor mode.

Full Software Emulation

```
check min[64K]
int16 reg[8]
int16 pc, sp
bool supervisor

loop
  insn = *pc++
  switch(insn)
    case mv a, b
      reg[b] = reg[a]

Trap and Emulate

User-space programs should execute directly.

1
insn
```

Fault:
Fault to hypervisor, which calls into guest OS.

Virtual Memory

Virtual to "Fake Physical" to Physical.
Working set: accesses \( \downarrow \) now
\[ \alpha \]
ws(\( \alpha \)) = pages accessed in the previous time \( \alpha \)
"footprint" = ws(\( \alpha \)) (all pages accessed)

Miss Ratio Curve

\[
\begin{array}{c}
\text{miss rate} \\
\text{memory size}
\end{array}
\]

Paging Example

FFFF00xxx - stack allocation, not filled

09000xxx - data

08000xxx - code/bin/aux

\( \Rightarrow \) MOV(*09000(hex), FAX)

PUSH EAX

- attempt 08000000

- fault on 09000+00

- allocate O page C

- allocate page table D

- resume

- attempt 08000000 \( \uparrow \)

- attempt 08000000

- fault or FFF00 111
On page fault, real CPU checks fake page table and translates fake phys to phys

Intel CPU is not virtualizable!

Virtualizing Unvirtualizable CPUs

1) Emulate all supervisor-mask code
   \[ \rightarrow \] JIT (optimization)

2) Paravirtualize
   hypercalls: modify PTI
   swap CR3
   set page fault handler

3) Hardware support

<table>
<thead>
<tr>
<th>mode</th>
<th>bitmask of where 00010000 to trap</th>
</tr>
</thead>
<tbody>
<tr>
<td>user mode</td>
<td></td>
</tr>
<tr>
<td>super mode</td>
<td></td>
</tr>
<tr>
<td>hyper mode</td>
<td></td>
</tr>
</tbody>
</table>
When VMware is running, it puts even the host OS into a sort of VM.