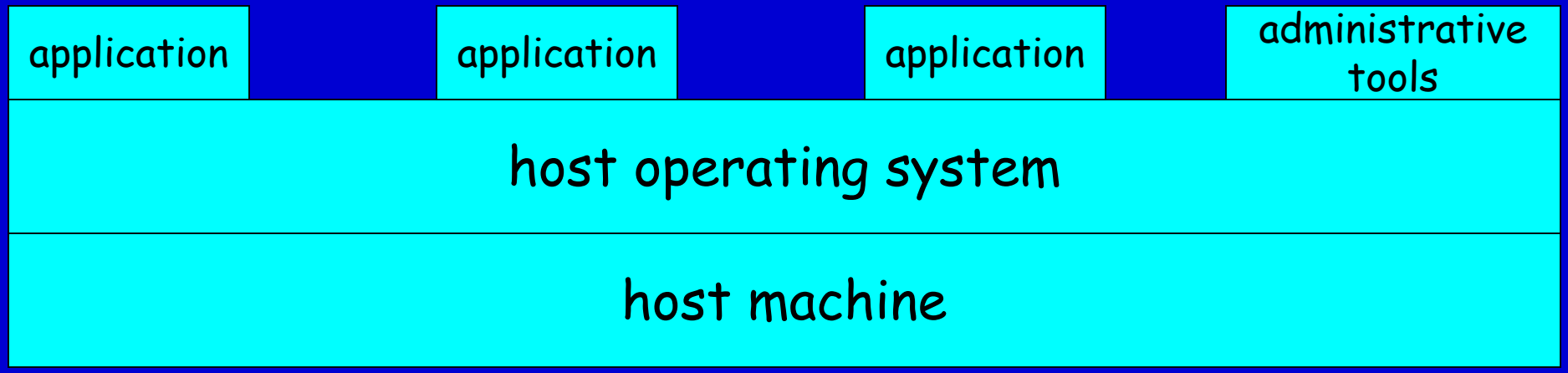
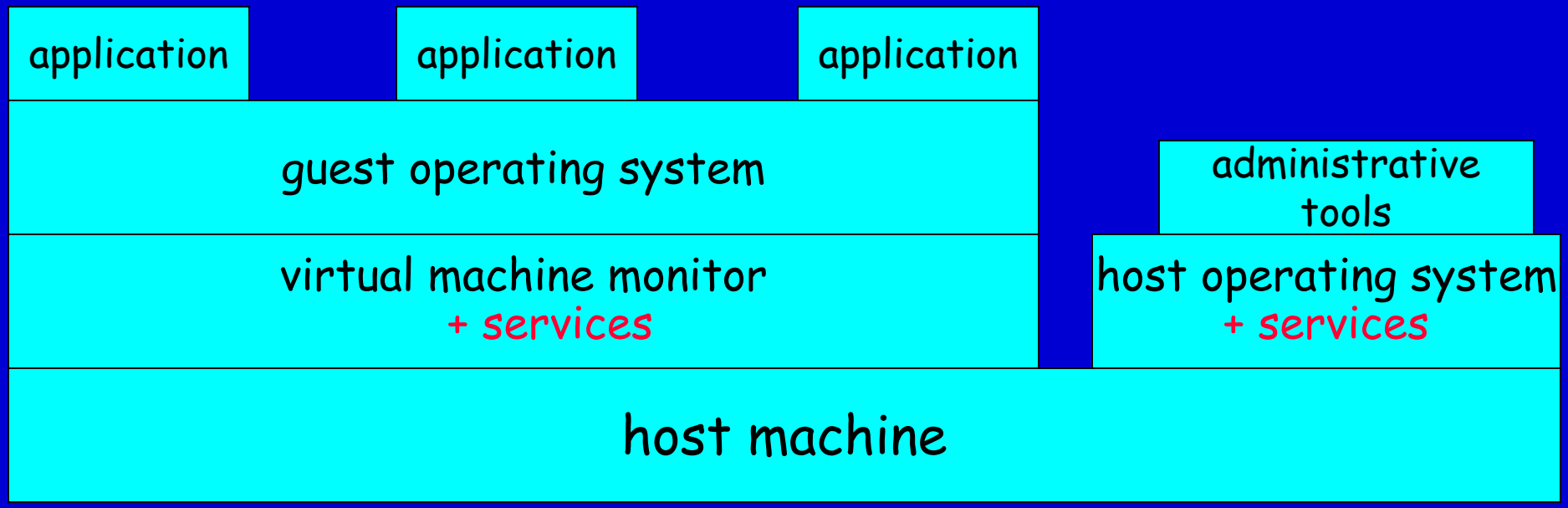
When virtual is better than real

Peter M. Chen
Brian D. Noble
University of Michigan

Standard system architecture



Virtual-machine system architecture



Benefits

- Services are protected from applications and guest operating system
- Services work for multiple OS versions and vendors
- Services benefit from unique abilities of virtual machines
  - e.g. create temporary virtual machines
  - e.g. communicate quickly to host
  - e.g. move virtual-machine state across network
  - e.g. encrypt virtual-machine state

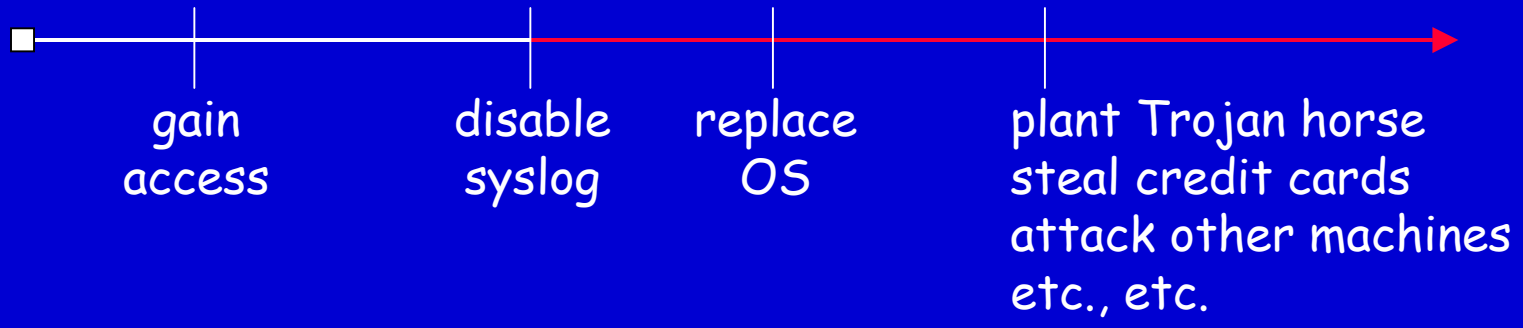
Challenges

- Overhead of running applications in virtual machine
- Semantic gap between events in guest OS and events in virtual machine

- Are there useful services that can work at virtual-machine level?
  - some services don't need to know about guest OS abstractions
  - some services can reconstruct semantic information common to "all" guest OSs

Secure logging

- Current systems log interesting events (e.g. logins)
  - vulnerable to OS compromise
  - may not anticipate relevant events
- Apply fault-tolerance techniques to log and replay complete execution of virtual machine
- Analyze any intrusion to arbitrary level of detail, even after point of OS compromise



Reducing log traffic

- Only log non-deterministic events
  - human input
  - interrupts
  - network messages
- Messages from cooperating hosts can be re-created instead of logged
  - remember message order
  - safely identify cooperating hosts
- If all hosts on LAN cooperate, only need to log incoming network traffic (at gateway)

Intrusion prevention

- Current systems block suspicious events before they compromise system
  - accuracy limited by fuzzy definition of "suspicious"
- Create disposable clone of the virtual machine, use clone to measure actual effect of suspicious event
- Enables destructive tests
- Open questions
  - semantic gap: VM detects OS-level effect?
  - what does original VM do while clone is testing event?

Intrusion detection

- Current detectors look for signs that system has been compromised
  - network-based detectors only see network packets
  - host-based detectors vulnerable to OS compromise
- Virtual-machine intrusion detector
  - monitor complete set of system events (CPU, memory, disk, keyboard, network)
  - monitoring continues even if OS is compromised
- Semantic gap: how to understand system events without re-implementing guest OS?

Environment migration

- Lots of ways to migrate state: thin clients, distributed FS, process migration, carry laptop
  - intolerant of latency
  - residual dependencies
  - require user intervention/management
- Virtual machines can encapsulate and move complete state of running computer
  - no OS changes
  - nothing to carry (or lose)
  - utilize remote computing resources

Migrating quickly

- Machine state can be very large: memory+disk
- Take advantage of sequential sharing patterns
  - logically one machine; no concurrent sharing of state
  - exploit pattern via DFS, shared memory techniques
- Not all state is needed right away
  - memory and disk working set size is visible
  - may successfully predict immediate needs
- Requires crossing the semantic gap
  - disk gaps are easy; physical blocks rarely remapped
  - memory is often remapped, via virtualized hardware

Other uses of encapsulation

- Fast migration depends on ability to do two things
  - encapsulate the entire state of a machine
  - identify critical state that will be needed soon
- Other potential uses for encapsulation
  - machine cloning for destructive hypothesis testing
  - encrypting entire machine state for arbitrary Oses
- Current encryption systems one-shot, incomplete
  - file system, swap space, secure RPC, …
- Can use encapsulation to guarantee all state
  - suspend virtual machine to (encrypted) disk
  - capture all network traffic below level of OS

Alternatives

- Add service to monolithic OS
  - trusts entire OS to be secure
  - trusts entire OS to be crash-proof
- Re-structure OS into isolated layers
  - requires OS modifications
  - similar tradeoffs to VM-based services: performance, semantic gap
- Language-level virtual machines
  - limited to applications written in specific languages

Conclusions

- Virtual-machine services have interesting potential ...
  - portable across different OSs
  - work despite OS compromise
  - clone, encrypt, transport state of entire computer
- ... and raise plenty of open questions
  - performance penalty
  - semantic gap