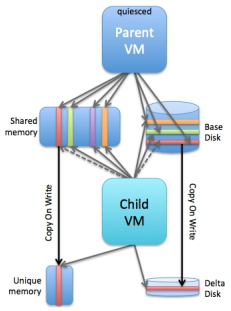
Автор:
22.10.14 04:22 -

22/10/2014 **Реклама:**

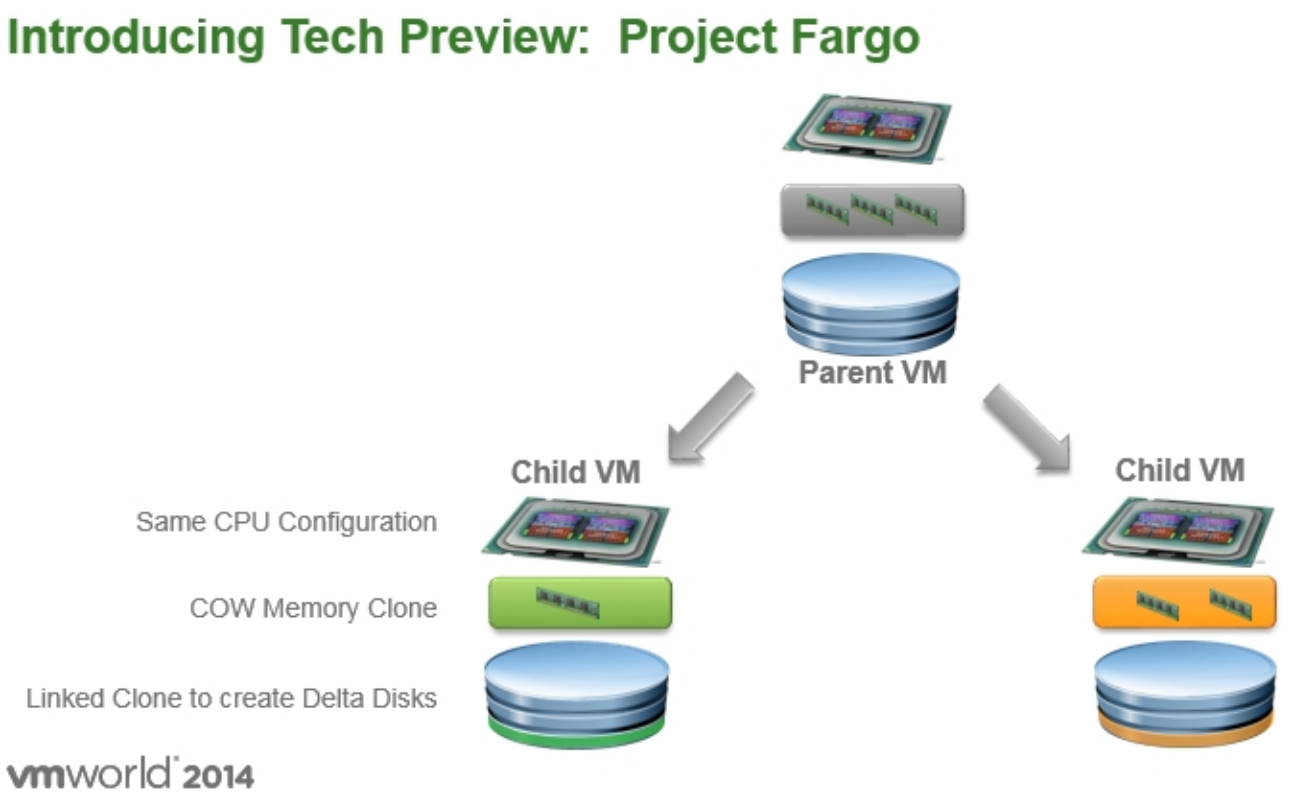
Пост:

Те из вас, кто следит за анонсами VMware VMworld 2014, которые были сделаны в Штатах, а затем в Европе, наверняка слышали о такой штуке как VMware Project Fargo (его прежнее название - VM Fork), которая позволяет очень быстро сделать работающую копию работающей виртуальной машины на платформе VMware vSphere.

Суть технологии VMFork такова, что "на лету" создается клон виртуальной машины (VMX-файл, процесс в памяти), который начинает использовать ту же память (Shared memory), что и родительская ВМ. При этом дочерняя ВМ в шаренную память писать не может, а для записи собственных данных используется выделенная область памяти. Для дисков аналогично - с использованием технологии Copy-on-write в дельта-диск дочерней ВМ пишутся отличия от базового диска родительской ВМ:



Автор:
22.10.14 04:22 -



A Fresh Approach to Application Publishing Using Project VMFork

Abstract

Vision:
'Project VMFork' is a platform technology for rapid VM cloning of already running VMs. Using this method, we can create VMs for Windows application remoting very quickly and with high resource efficiency. Preliminary testing indicates that this approach can equal or surpass traditional RDSH based approaches. Project VMFork based application publishing brings all the power of VM containers to each application session, offering low cost and superior functionality.

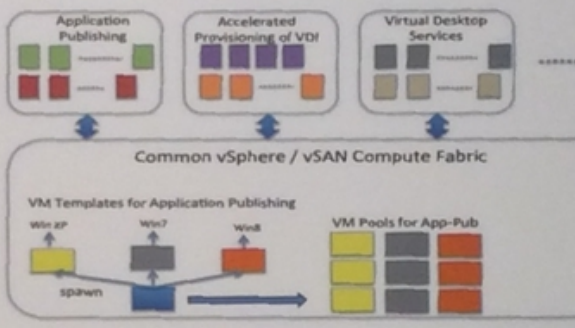
Technology:
- Leverage Project VMFork based VM cloning for on-demand application publishing
- Special 'secret-sauce' In-Guest techniques that help leverage Project VMFork include pre-launching a catalog of applications
  - Prepare a 'lean' Windows image – disable unnecessary services and components
  - Install catalog of applications
  - Log in with local-user account
  - Pre-launch all applications using a startup-script
  - Pre-load all application pages into memory using special utility that touches each code page and then locks the process into memory - allowing child VMs to avoid ever loading any code pages off disk
    - Limits I/O on Child VMs
    - Limits Copy-on-Write Memory operations
    - Limits CPU associated with I/O and memory operations
  - Suspend Processes from running to ensure no-interactions between launched apps and chosen application in the resulting Project VMFork 'child VMs'
    - Avoids any possible negative interactions between the catalog of pre-lau and the one which the user will select to use
    - Eliminates background CPU activity by the large number of launched app
  - Upon user requesting a Remote Application
    - Broker can 'Fork on Demand' – few seconds to running VM that has all ap launched in pre-logged-in session.
    - User is 'Reconnected' to existing disconnected session using the local-use but broker passes in that user's actual name/password securely
    - Acquire network token with a background 'Net Use' and bind token to se application for user impersonation.
    - Resume selected application and maximize

vSphere Modifications:
- Leverage coming 'Project VMFork' feature of our 2015 platform release
- Leverage new vCenter APIs coming in 2015 platform release

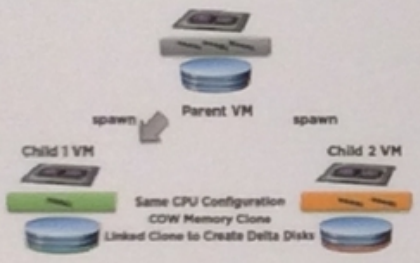
Performance Results:
- Fully functional Windows VMs with running applications running in < 4s Mass of running App-Pub VMs with launched application – 500 in less than 5 minutes
- 120 sessions per 2 socket 8 core host with approximately 30% less CPU than number of RDSH sessions
- Modestly better memory utilization than RDSH
- I/O reductions using advanced application pre-loading techniques

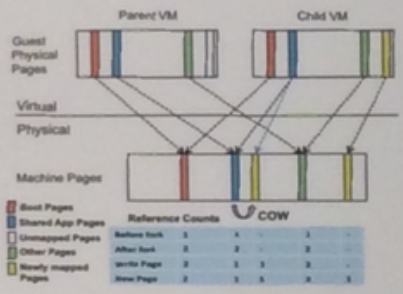
Project VMFork Architecture

- Different VM pools of desktops and application publishing can leverage Project VMFork. For the high memory sharing efficiency, use a single VMFork parent for each host in a cluster
- Pools of child VMs based on each template can host a different OS or different set of applications
- Project VMFork maintains powered-on parent template VMs and a set of pre-registered but powered-off children VMs which cause a live Fork to occur when they are powered on or 'reset'
  - Children are pre-registered to VirtualCenter (VC) to remove registration from the critical path of deployment
  - Each pre-registered child VM has its own VMX file
- Upon request for a published application the children VMs are instantly powered on with unique IP/MAC and quickly configured with 'finishing touches' for the application requested



Fast VM Instantiation



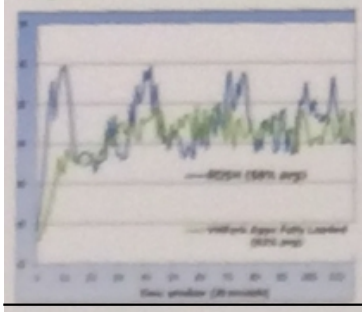


- **Prepare parent VM:**
  - Bring parent to desired state with launched apps fully loaded into memory then issue tools guest RPC call to quiesce parent and marks pages as COW (Copy-on-Write)
- **Prepare child VM:**
  - Linked clone to create redo log, prepare VM config file for customizing children (e.g., IP, MAC)
- **Spawn child:**
  - Power-on VM, resumes from the parent's state, waits for incoming app-pub session before resuming and maximizing the selected application for remoting back to user

- Disable large page sharing & optimize transparent page sharing to quickly id sharing opportunities among COW pages
  - Mem.ShareUpdatePeriod: change from 30 to 5
  - Mem.ShareScanTime: change from 60 to 10
  - Mem.ShareScanGHz: change from 4 to 32
  - Mem.ShareRateMax: change from 1024 to 32768
- VMs for App-pub are short running so COW is limited and VMs are destroyed end of the session

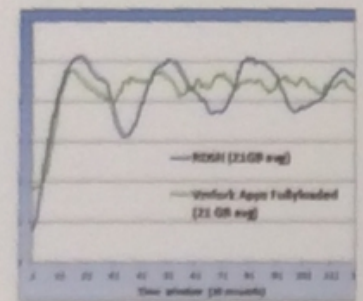
CPU Performance:

Project VMFORK APP-PUB vs. RDSH



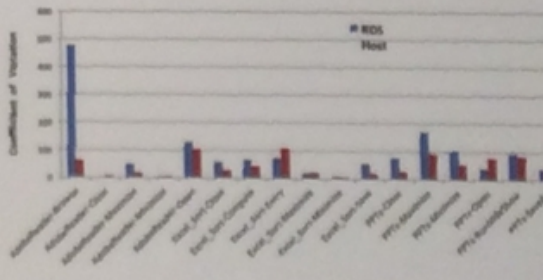
Memory Utlization:

Project VMFORK vs. RDSH



Execution Variance:

Project VMFork vs. RDSH



- Project VMFork based App-publishing workload on View Planner
  - 100 sessions running short duration single application workloads
  - Project VMFork based VM sessions used about 20% less CPU
  - Memory consumed was about 22GB for both VM and RDSH designs

- Project VMFork based App-publishing workload on View Planner
  - View Planner average execution times were very similar between two ap models – slight advantage to VMFork – 21 vs 19.
  - Std. Dev of execution times was much better on Project VMFork design, on very short tasks like Adobe Reader Browse (5x advantage to VMFork)

Автор:
22.10.14 04:22 -