OpenXT Architecture

Christopher Clark
BAE Systems

OpenXT Summit, 7-8th June, 2016

Presenter: Christopher Clark

Xen affiliations, since 2003:

- **BAE Systems**, OpenXT -- as of January 2016.
- { non-Xen work, 4 years }
- Citrix
- XenSource
- Intel
- Cambridge University

Projects: **OpenXT** - XenClient XT - XenClient - XenServer - Xen hypervisor

Roles:

- **Interoperability Architect**
- Release Manager; shipped first GA releases of XenClient and XenClient XT.
- Principal Engineer
- Graduate Student

Most recent work: **Governance** for the OpenXT Project.

Outline

- Introduction to OpenXT
- Rapid tour of components and technologies of OpenXT
    - NOUNS : introducing items that will likely be referenced during the rest of the Summit.
- Distinguishing properties of OpenXT
    - Consequences of the structure of our collection of nouns

OpenXT: hardened client virtualization platform

IT IS COMPLEX.

“OpenXT is a development toolkit for hardware-assisted security research and appliance integration. It includes hardened Linux VMs that can be configured as a user-facing software appliance for client devices, with virtualization of storage, network, input, sound, display and USB devices. Hardware targets include laptops, desktops and workstations.

OpenXT stands on the shoulders of the Xen Project, OpenEmbedded Linux and Citrix XenClient XT. It is optimized for hardware-assisted virtualization with an IOMMU and a TPM. It configures Xen network driver domains, Linux stub domains, Xen Security Modules, GPU passthrough, Intel TXT, SELinux, and VPNs. Guest operating systems include Windows, Linux and FreeBSD. VM storage options include encrypted VHD files with boot-time measurement and non-persistence.

OpenXT has evolved from cross-domain endpoint virtualization to an extensible systems innovation platform, enabling derivative products to make security assurances for diverse hardware, markets and use cases.”

… BUT IT EXISTS. IT WORKS. IT IS SIGNIFICANT.

OpenXT nutshell

- Core technology: **Virtualization**. Xen.
- **Client** focus: endpoint execution environment.
  - On laptops, desktops, workstations, remote headless and beyond.
- **Security**
  - Architected to support strong assurances rooted in hardware.
- **Toolkit**: extensible
  - OpenEmbedded toolchain.
  - Service VMs and APIs.
- **Open Source**: permissive licensing,
  - Suitable for diverse use cases.

What makes OpenXT different?

- Our **community**:
    - **Depth and diversity of experience, direction of focus and level of expertise.**
- **Sophistication** of the software:
    - *An integrated system* of many complex software components.
    - Derivative projects have been successfully Certified.
- **Security** focus, for stronger assurances.
- **Client** deployment use cases.
- **Open Source**, OpenEmbedded build and toolchain.
    - Participation in upstream communities.
    - Commitment to support downstream projects.

Architecture Goals for OpenXT

- Support for client use cases
  - Compatibility with modern hardware and operating systems.
- Raise the security of OpenXT and upstream projects
  - Disaggregation of privilege and strong enforcement of isolation.
  - Harness the hardware platform capabilities.
- Support upstream engagement
  - Design for upstream acceptance.
  - Promote work into upstream projects to benefit wider audiences.
  - Participate upstream. Support, guide and advocate for technical changes for our use cases.
- Support downstream projects

See also the OpenXT Platform Properties and Layers Governance Document.

Build Product

A current build of OpenXT today will give you:

- An installation disk, that will provision the Xen hypervisor and a collection of helper VMs to provide a hardened desktop environment suitable for installing client VMs to run Windows or Linux desktop workloads, and paravirtualized device drivers for use within those client VMs.

Extensibility of this system:

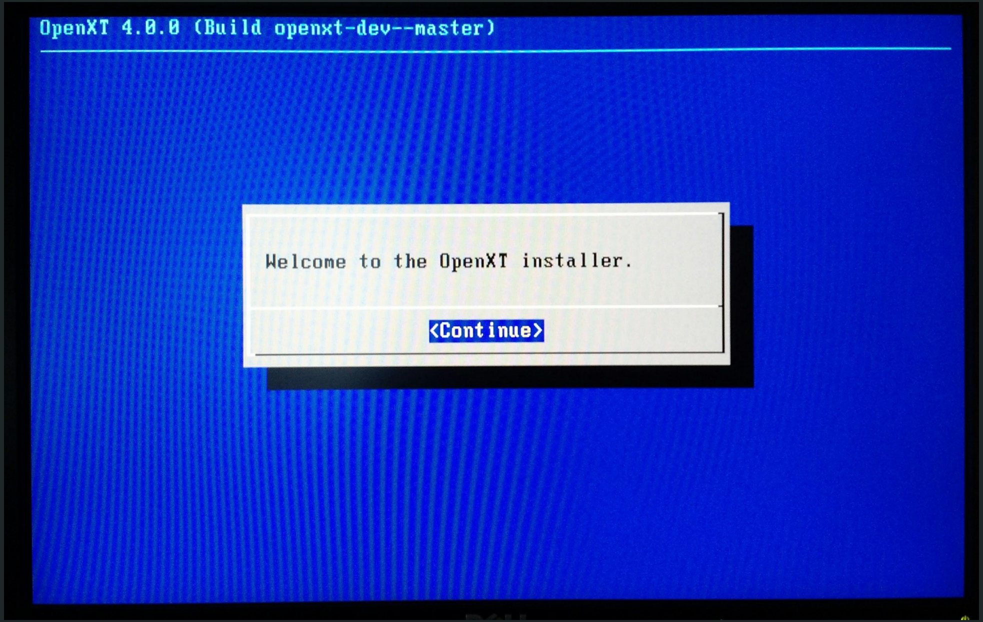
- At software compile and build time: customizable via OpenEmbedded Layers.
- At deployment time: API support for third-party Service VMs.

OOBX

Hardware requirements:

VTx, VTd (optional), TXT (optional), TPM (optional)
Fully updated BIOS.

- Walk through simple questions.
- Installs to the machine hard disk.
- Claims ownership of and provisions the TPM.

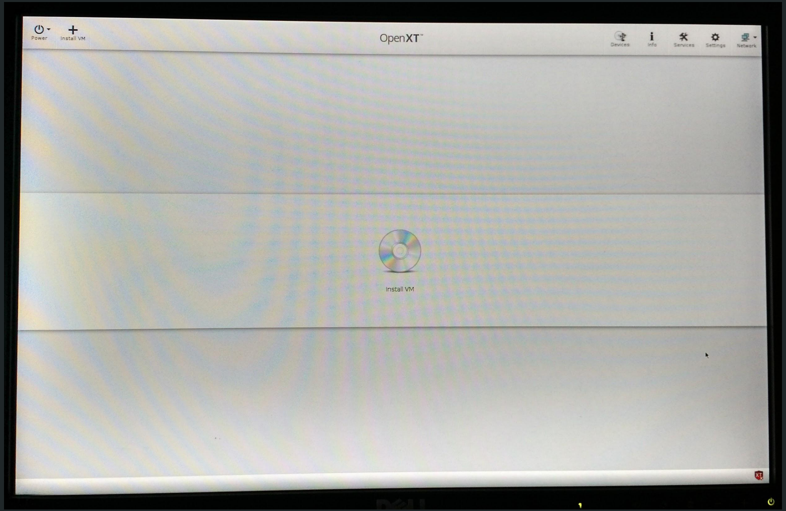


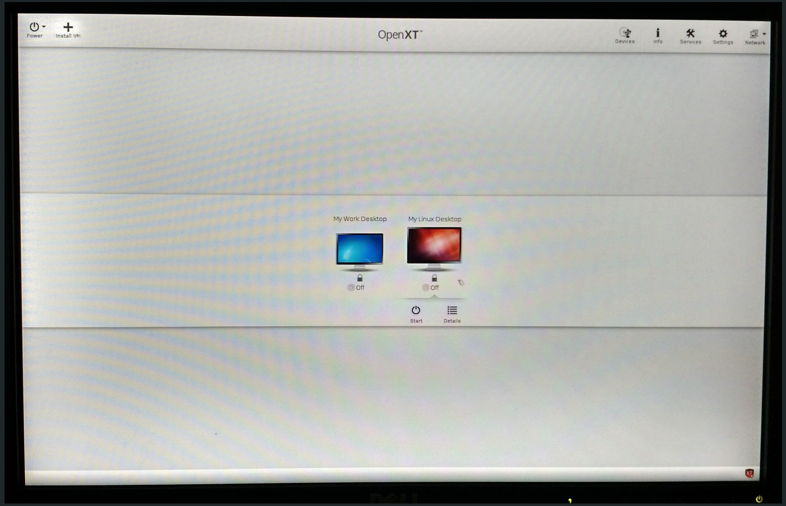
```
OpenXT 4.0.0 (Build openxt-dev--master)

Welcome to the OpenXT installer.
```

Installed System

First boot after fresh install:





and then with a couple of VMs added, Windows and Linux desktops.

The Hardware Platform

Utilizes major **Intel platform technologies**:

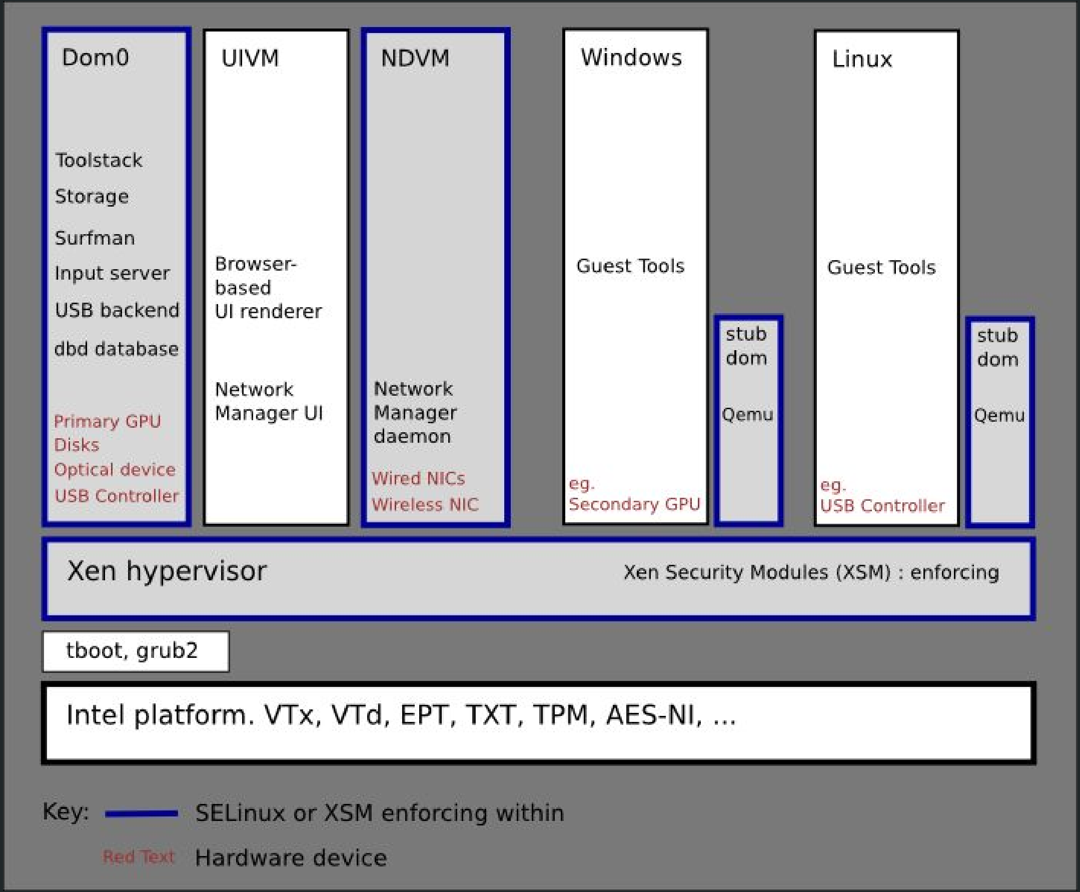
- VTx, VTd, TXT, EPT, TPM, AMT, AES-NI, ...
- There will be more (processor cache partitioning, SGX, ...).

Also Intel client technologies: graphics, wired and wireless networking, and audio.

ARM? Not currently.

- Potential exists.
  Some OpenXT community members looking into this.

System Architecture



Xen hypervisor and Qemu device emulator

Numerous modifications vs Upstream Xen + Qemu. Motivated by our secure client use cases.

- Xen Security Modules (XSM): Enabled and Enforcing.
- Suspend, hibernate and power management changes.
- v4v interdomain communication protocol + firewall.
- Modifications to blktap2 storage support.
- Cosmetic fixes to correctly display Windows boot graphics.
- ACPI and SMBIOS support for laptop vendor customizations.
- Hardware quirk fixes.
- Setting cache attributes on mapped memory for video support.

etc., etc...

The delta vs upstream is decreasing as upstream capabilities improve with our involvement and with some former XenClient features being removed from OpenXT code.

System Virtual Machines

OpenXT uses OpenEmbedded to build several Linux-based operating system images.

- Dom0 : privileged control domain.
- Stubdoms : Isolated device emulator domains, one per guest Virtual Machine.
- UIVM : **U**ser **I**nterface **V**irtual **M**achine: Runs the host user interface accessible on the local console.
- NDVM : **N**etwork **D**evice **V**irtual **M**achine: Operates the physical NICs. Contains NIC device drivers.
- OpenXT Installer: a bootable ISO to install OpenXT onto a computer.

Interdomain communication channels

- Xen's grants and events
  - Grants share, transfer or copy memory pages between VMs. Copying is now favoured.
  - Events are virtualized interrupts.
  - Typically combined for use in “shared producer-consumer rings”.
- Xenstore
  - A hierarchical key-value store with simple access control.
  - Typically used for communicating identifiers and small bootstrapping values.
- **v4v**
  - An OpenXT technology.
  - High-performance data transfer with simple mappings to Windows and Linux I/O primitives.
  - Firewall rules enforced.
- dbus : proxied over v4v
- Network : optional, according to policy and configuration

Isolation enforcement

- Standard Xen mechanisms, paravirtualization of guest devices
- VTd to securely assign and confine PCI devices to within a VM
    - Network interface cards
    - Graphics display cards
    - USB controllers
- Stubdomains
    - OpenXT uses Linux-based stubdomains with SELinux confinement
        - http://www.slideshare.net/xen_com_mgr/linux-based-stubdomains
- Mandatory Access Control
    - SELinux policies tailored to our software. Enforcing.
    - Xen Security Modules (XSM). FLASK architecture. Enforcing.
- Disaggregated network stack via helper VMs
- Read-only root filesystem with reset-on-boot temporary write space for stubdoms, NDVM, UIVM.

Qemu and OpenXT stub domains

OpenXT differs from other Xen-based projects:

For each guest VM, uses a stub domain that contains:

- Linux operating system with limited filesystem, reset at boot
- Qemu device emulator
- SELinux with a targeted policy, enforcing.

This **isolates each Qemu** device emulator instance.

Qemu's attack surface is large due to the many functions that it performs and the complexity inherent in device emulation.

In OpenXT, compromise of Qemu does not compromise the host. **XSA attacks defeated by architecture.**

Toolstack

User-space application code glue, control plane, mainly in Haskell and OCaml.

- Xen Manager
- xenvm
- xec CLI tool

Libxl is the upstream Xen Project C-language library for driving Xen mechanisms.
A project is under way to migrate the OpenXT toolstack to use libxl and move much of the toolstack out of Dom0 into an isolated deprivileged domain.

This work will also assist development and use of alternative OpenXT toolstacks.

On-disk layout

Partition and file-based encryption. Not Full-Disk Encryption.

- Boot, Root : Read-only standard OpenXT system software images.
- Config : Encrypted partition. Key secured via Measured Launch.
    - Configuration settings of the system. (eg. "/etc").
    - VM metadata files and VM encryption keys.
- Storage : Individually encrypted disk image files of VMs.
- Log : Dedicated bounded space

Integrity is protected by measurements and secure boot.

Tampering with the system software will be detected during measured launch. The keys to access the user data on the system will not be available. The system will allow boot and indicate the error to the user. An administrative password will be required to unlock access and decrypt the data.

Storage path

Prior to installing PV drivers in the guests: Qemu device emulator provides access to the VM's disk image.

Once PV drivers are installed: blkfront communicates with blktap in dom0 over shared grant-event rings.

OpenXT uses blktap2, as opposed to using qemu, for the high-performance production data path.

- blktap2 is considered "un-upstreamable" into mainline Linux
- blktap3 exists as a project and is entirely user-space. Requires Xen grant copy code in Linux kernel.
- Some Xen project participants are using qemu as their primary storage data path.
    - OpenXT has an architectural choice to make and path to pursue here.

Disk encryption is performed with LUKS. AES-NI accelerates encryption and decryption data paths.

Measured Launch

OpenXT uses Intel TXT in a measured boot sequence to securely verify the integrity of each of the system software components started during system launch.

- Measurements are stored in the TPM.
- Measurements include: Xen binary, Dom0 kernel, kernel command lines, Dom0 initrd, Dom0 rootfs.
- Uses grub version 2, with a fixed, measured grub command line. Interactive grub prompt is disabled.

The measurements of all components must be correct to acquire the decryption key to unlock access to the host platform configuration and all guest VMs.

Notable components:

- Tboot is the Xen launch verifier for TXT.
- TrouSerS, TCG Software Stack. User space software that drives the TPM via a Dom0 kernel device.

Network architecture

In the default OpenXT configuration:

All of the physical PCI network devices are assigned to a **single N**etwork **D**evice **V**irtual **M**achine (**NDVM**) and **isolated using VTd**.

The NDVM contains all the network device drivers and exports access to the networks via virtual interfaces into the guest VMs. Bridging or routing guest networks and physical networks happens in the NDVM. Important: **Dom0 is not part of the guest networking datapath.**

User control of networks is via the Network Manager in the User Interface VM. This connects to the network-manager daemon via dbus proxied over v4v.

This **isolation of the network device drivers** is one of the key distinguishing properties of OpenXT.
A compromised network device driver only compromises the NDVM and not the rest of the system.

Network architecture (cont'd.)

OpenXT also supports alternative network configurations running **multiple NDVMs**: one per NIC.
Configuration instructions are in the project documentation.

This stronger configuration **enables VTd-enforced isolation between physical networks**.

It costs additional resources (RAM, CPU) to run additional NDVMs, and can alter maximum throughput and packet latency across networks via the host, but it enables insertion of protected middlebox VMs into the cross-network path.

Network architecture (cont’d.)

The OpenXT platform supports development of "NILF-VM"s: **N**etwork **I**n-**L**ine **F**unction **V**irtual **M**achines.

This is to support third-party software integration.

An example:
A VM that provides encapsulation of traffic via a VPN, for interposition in-between a guest Windows VM and the NDVM assigned to the physical device NIC.

This isolates VPN credentials from both the guest Windows software and the network device driver.
The unencrypted traffic data is never accessible by the NIC hardware or the network device driver.

Secure Keyboard and Mouse Input

The input server runs in Dom0.

- Monitors udev for new input devices
  - Claims HID devices: keyboard, mouse, touchpad
- Demultiplexes input events
- Directs input events to the intended recipient VM
  - Including the UIVM, showing the local console UI
  - Prevents input snooping by other VMs
  - Enforces platform screen lock
  - Performs secure password capture for Dom0 : UIVM cannot snoop.
- Scales mouse movement for guest VMs running at different resolutions

Graphics

Onboard Intel Graphics Device is assigned to Dom0.

Xen and VTd enforce isolation such that Dom0 maintains control.

Guest VMs have a dedicated full-screen 2D framebuffer each.

These are isolated by Xen. Guest VMs cannot access the framebuffers of other VMs and screen-scraping is prevented.

Selection of which framebuffer to render to the display is controlled by Dom0.

Secure keyboard input sequences direct Dom0 to switch the display between VMs.

Secondary PCI Graphics Devices can be assigned to a guest Virtual Machine.

This enables high-performance 3D graphics capabilities within that VM with VTd-enforced isolation between VMs.

- “Surfman” is the component that manages the onboard graphics device.
- A community proposal exists for a new graphics and input implementation.

USB

A host USB controller can be assigned either to the OpenXT platform or dedicated to a single guest VM.

When a USB controller is **assigned to a single guest VM**:

- All devices physically connected via the controller will be accessible only to that VM.
- It uses the guest VM’s own USB controller device driver.
- The guest VM is responsible for its own USB device policy enforcement.
- Isolation between the USB devices and other guest VMs is stronger and VTd enforced.

USB (cont'd.)

When a USB controller is **assigned to the platform:**

- Connected devices are governed by **OpenXT USB policy** in effect for the **type of device**.
- **Isolation is not VTd enforced. USB management software runs in Dom0.**
- Individual USB devices can be assigned to different VMs or assigned to the platform.
  - eg. A USB optical disk device can be attached to the platform. Inserted optical media will then be delivered to the VM on the display at the time of media insertion.
- VMs can also have policy settings to prevent assignment of specific USB device classes.
  - Input : Usually connected to the platform. Can be assigned to a guest VM if required.
  - Storage : Can be directed into guest VMs or made accessible to Dom0 for developer or support use.
  - Networking devices and 3G modems: not subject to NDVM isolation.
  - Isochronous : Webcams challenged the PV-USB implementation, but now work fine within guest VMs.
  - Misc: iPad firmware updates, toy rocket launchers, sound cards...

USB (cont'd.)

OpenXT has its own mature, capable, paravirtualized USB device driver implementation.
It is different from other Xen platform PV-USB systems.

Upstreaming is desirable; effort required.

Optical Media

- Inserted media is connected to the VM currently displayed on the graphical console at the time of insertion, if policy permits access. Intuitive behaviour for interactive use.
- Media access can be assigned read-only, or read-write to allow burning disks.
- Guest VM interface is via the Qemu device emulator.
  Interprets ATAPI commands.

In-guest Tools

Paravirtual Drivers for modern Windows and Ubuntu.

- Network
- Disk
- Input (Mouse, Keyboard)
- USB
- Audio

Have been WHQL certified by Microsoft for previous releases. (Current status?)

All packaged into a standard Windows software installer.

Accessed in-guest via a virtual optical media device.

“The Switcher Bar” : a soon-to-be-removed in-guest user interface tool to interact with the host platform.

Distinguishing Properties of OpenXT

A maintained list of these is at: https://openxt.atlassian.net/wiki/display/CS/Gov2%3A+OpenXT+Platform+Properties+and+Layers

- Integrity of operating software assured via measured launch and hardware-rooted enforcement of isolation between components.
- Protection of storage, of system configuration data and user data at rest.
- Protection of communications, with isolation between network device control software, encryption software with network credentials, and user software.
- Protection of concurrent user software execution environments.
- Protection of user input.
- Protection of graphical output, with support for high-performance 3D desktops.
- Policy-based assignment and confinement of hardware peripherals.
- Compatibility with modern computer hardware and contemporary operating systems.
- Architected and licensed to support commercial derivative software.

Where we work on OpenXT

- Public email mailing list
- Monthly Community telephone call open to all
- Confluence public wiki
- JIRA issue tracker
- IRC #OpenXT on freenode
- Engaging with upstream projects
- Governance documents are on the wiki

http://openxt.org

EOF

Any questions?

-**xt**opher