Xen Security Weather Report
May 2018

Lars Kurth
Community Manager, Xen Project
Chairman, Xen Project Advisory Board

lars_kurth


SOFTLAYER
an IBM Company
ORACLE
CLOUD
Amazon Lightsail
kt
CITRIX
SUSE
ORACLE
amazon
web services
Tencent 腾讯
inspur 浪潮
rackspace
the open cloud company
DATAPIPE
HUAWEI
Alibaba Cloud
aliyun.com
ZEDEDA
NXP
XZD
XENZYNQ
DISTRIBUTION
DORNERWORKS
cisco
Bitdefender
ais
epam
atihita
GlobalLogic
Bromium
aggios
Xen
Project
OpenXT
QUBES OS
A REASONABLY SECURE OPERATING SYSTEM
AFRL
THE AIR FORCE RESEARCH LABORATORY
CRUCIBLE
ais
XILINX
TRIITUS
CORPORATION
BAE SYSTEMS
galois

Xen Project Security Status

Meltdown/Spectre Updates

Meltdown (SP3)

- **XPTI:** Recommended mitigation
- Followed by XPTI performance improvements – more to follow

Spectre (SP2)

- **x86:** By default, Xen will pick the most appropriate mitigations based on compiled in support, loaded microcode, and hardware details, and will virtualise appropriate mitigations for guests to use.
  - Command line controls via the spec-ctrl command line option are available
  - 10% net/disk performance improvements in some production scenarios
  - 50s boot time improvement on some servers

Meltdown/Spectre Updates

Spectre (SP2)

- **Arm32:** Mitigation for Cortex-A15, Cortex-A12, Cortex-A17 are present in Xen 4.7 and later (requires updated firmware)
- **Arm64:** SMCCC 1.1 based mitigation for Cortex-A57, Cortex-A72, Cortex-A72, Cortex-A75 for Xen 4.10 and later.

Spectre (SP1)

- Attacker needs a suitable "gadget" running inside of the Hypervisor
  GPZ use the eBPF engine to execute a gadget written by them, designed to 'leak' the maximum amount of information
- Initial analysis: Xen has nothing similar to eBPF and no other gadgets
- Static analysis by large vendors in the community has also not yet shown anything

Security Functionality is now widely used

Disaggregation

- **Desktop:** OpenXT, SecureView, QubesOS
- **Embedded:** Virtuosity (Q2-Q4 2018), Crucible
- **Automotive:** EPAM Fusion, GlobalLogic Nautilus

Virtual Machine Introspection

- **Server Virtualization:** Bitdefender HVI, Zazen, AIS IntroVirt
- **Security Software:** Cuckoo Sandbox, DRAKVUF

Live Patching

- **Server Virtualization:** Citrix, Oracle, Huawei
- **Cloud Computing:** AWS, Oracle, Huawei, IBM Softlayer, Rackspace, … ~50% of pre-disclosure list

These technologies are well understood by a critical mass of Xen Project upstream maintainers and committers.

A note on Disaggregation in upstream

Support Status

- **Driver Domains:** Security supported with caveats (stemming from PCI Passthrough caveats)
- **Device Model Stub Domains:** Vulnerabilities of a device model stub domain to a hostile driver domain (either compromised or untrusted) are excluded from security support

Missing upstream Pieces

- **Upstream Multi-Domain Build Support:** OpenXT, various vendors, … do their own different things
  ➜ ViryaOS could be the way forward (see lists.xenproject.org/archives/html/xen-devel/2018-05/msg01097.html)
- **Lack of upstream integration testing:** looking for contributions or commitment to test RCs
  ➜ see https://lists.xenproject.org/archives/html/xen-devel/2018-05/msg00976.html
- **Linux stub domains:** waiting for Invisible Things Labs contribution

Upstream support could be better if there was more active engagement from down streams

This is generally the case for security features with the exception of LP and VMI
➜ more leadership/engagement from security community needed

Security Functionality is now widely used

Measured Boot: TPM/TXT (x86) & OP-TEE (Arm)

- **Desktop:** OpenXT, SecureView, QubesOS (optional via Anti Evil Maid)
- **Embedded:** Virtuosity (Q2-Q4 2018), Crucible
- **Automotive:** EPAM Fusion – actively working on OP-TEE integration with Xen
- **Cloud Computing:** Oracle (in progress)

Not well understood by upstream maintainers and committers.

XSM:FLASK

- **Desktop:** OpenXT, SecureView
- **Embedded:** Virtuosity, Crucible
- **Automotive:** EPAM Fusion

Not well understood by upstream maintainers and committers. There is a desire to get XSM to a point where we would like XSM to be supported and could recommend average users to enable; possibly even having it on by default.

The Problem with XSM:FLASK

Disaggregation involves breaking up Dom0 into several domains

- These perform actions normally reserved to Dom0 ➜ meaning more permissions are needed
- XSM:FLASK is then used to relax permissions of domains that would be DomU's

Risks from the viewpoint of the Xen Project

- High risk to enable XSM with a policy that makes the system less safe by allowing 'normal' domains to do things they weren't able to do before
- Using XSM requires expert capability that most users will not have
- Need a way of guaranteeing that XSM + the "default policy" is at least as safe as no XSM
- Also see XSA-77 (Disaggregated domain management security status)

Possible Solutions

- A test based approach ➜ cumbersome and does not protect users modifying the policy
- An approach that allows starting privileged domains and uses XSM:FLASK to restrict permissions
- In other words, regular domains would be excluded from from the scope of XSM:FLASK
- Not alone sufficient to become a supported feature

Security Process Changes: Support.md

Support.md

xenbits.xenproject.org/docs/unstable/support-matrix.html

- Formalize support and security support status for Xen Functionality
- This highlighted some gaps, such as VMI, XSM, XSM:FLASK
- Some partially security supported features such as Driver Domains, Device Model Stub Domains, Device Passthrough have raised concerns

What is needed for Features to be (security) supported

- Functional completeness, sufficient automated testing, interface stability ➜ supported
  - Automated testing via OSSTEST can be waived if there is a commitment by a community member to run their own automated tests on Xen RCs
- In most cases this is sufficient to be security supported
  - But: in cases where the security team does not have the capability to fix issues, we need commitment from outside experts to work with the team under embargo in a timely fashion
    ➜ Example: ARINC635 scheduler

Security Process Changes: Other

Become a CNA

- Applied to MITRE after a an informal agreement in mid 2017
- Responsibility for awarding CNA status for FOSS projects has been moved to DWF (so far unresponsive)

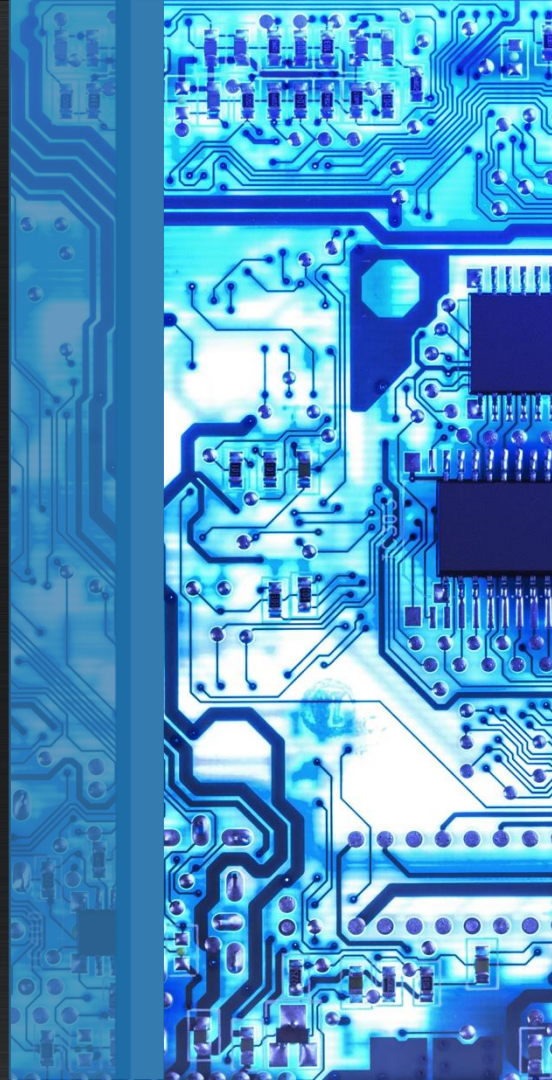
Security Process Review

- Performed analysis of Security Process with some recommendations
- Available for review at: lists.xenproject.org/archives/html/xen-devel/2018-05/msg01127.html

Observation

- Very few Security/Embedded companies are on the pre-disclosure list despite qualifying: Only AIS, BitDefender, Invisible Things Lab are on the list
- I do not understand whether there is a valid reason or this is simply an oversight

Ongoing and Future Projects



Panopticon: Resilience to Future Side Channel Attacks

Basic Idea: Remove Sensitive Data from the Hypervisor

- Follow-up on Spectre/Meltdown issues to reduce the risk of data leakage
- **The problem:** all state from all VMs is currently mapped into the hypervisor
  ➜ so an attacker can see secrets belonging to other VMs.
- **Solution:** reorganise the mappings so the absolute minimum amount of data is mapped
  ➜ an attacker can't access data pertaining to other VMs.
  - Get rid of Directmap, etc.
  - Per domain heaps
  - Reduce non-relevant data that is mapped into Hypervisor context

This will be a hot topic at the Xen Project Developer Summit in Nanjing, China

Evolving x86 PVH Support

Status at release of Xen 4.11

- PVH DomU (no passthrough, QEMU not required) – supported
- PVH Dom0 – experimental
- PVH Shim (backwards compatibility) – supported primarily because it has received significant testing and was used as Meltdown mitigation
- PV / HVM code path separation: progress at around 60% through the code

Future

- Add PCI passthrough support for PVH DomU
- Performance tuning of PVH Dom0 (hypercall performance of PVH guests compared to PV)
- Complete PV / HVM code path separation
  - KCONFIG configuration for PV and PVH/HVM modes (CONFIG_PV and CONFIG_HVM)
    ➜ significant attack surface reduction
- Make all relevant components supported

x86 code size: what may be possible

Today (Coverity based on standard x86 build)	K SLOC
Xen x86 with everything enabled	237
QEMU Upstream x86 / QEMU Traditional x86	1,005 / 125

Possibilities, once PVH work has been completed

Estimates (based on manual inspection)	K SLOC
x86 PVH for Intel only (no AMD) No Server features No XSM, No VMI	128
Additionally remove __init code (which get discarded once hypervisor is fully set up) and additional components which are not needed for a more static system as expected in embedded/automotive applications	110

Substantial code size and thus attack surface reduction is possible, but would require significant effort.
A more comprehensive study on x86 code size would be needed.

Automotive/ Embedded Projects: Mixed Criticality Workloads

wiki.xenproject.org/wiki/
Category:Safety_Certification

Major ongoing/upcoming contributions

Items I am aware of (not a complete list)

- PV drivers: input, sound & DRM (EPAM)
- Xen OP-TEE support (EPAM)
- Co-processor (GPU) sharing framework (EPAM)
- Hard real-time support (EPAM)
- Power Management & HMP (Aggios, XILINX)
- Startup latency: Boot multiple VMs in parallel from Device Tree (XILINX)
- RTOS Dom0 / Dom0-less system (Multiple)
- Code size reduction for Safety certification (Multiple)
- Inter-VM communication primitives for hypervisor mediated data exchange (BAE)
- Virtual TPMs for Xen in OpenEmbedded meta-virtualization (BrainTrust)

Arm Hypervisor code size: starting point

Full ARM 64 build with **standard configuration** (via make cloc) is **~58.5K SLOC**

Excludes:

- ACPI (not enabled in default config)
- Dom0
- Key Dom0 components
- Toolstack

Renesas Rcar-3 Kconfig is **~48K SLOC** today.
See lists.xenproject.org/archives/html/xen-devel/2018-04/msg01453.html for a patch under discussion.

Potentially Feasible:

A smaller Xen configuration of around **45K SLOC** may be achievable,
by disabling some core functionality
→ not clear whether desirable and the effort required

Cost Example: DO-178C, **per 10K SLOC**

DAL E (0.11 h/SLOC): **~0.6** man years … ASIL-A
DAL C (0.20 h/SLOC): *~1* man years … ASIL-B/C
DAL A (0.67 h/SLOC): **~3.3** man years … ASIL-D

Hours for vendor with certification experience
Based on data from Dornerworks

Safety Certification: Straw plan

MISRA Compliance

1 Identify compliance partner that is willing to work with the project → PRQA

2 WIP: Formalize relationship between vendor and the project

3 Iteratively address compliance issues within the Xen Project community: start with potentially controversial and high impact issues.

4 Complete MISRA compliance work for majority of issues.

Dom0

RTOS (e.g. FreeRTOS) as Dom0, and/or Dom0-less stack with minimal management tools.

Lead Community Members

- EPAM, XILINX
- Dornerworks and Star Lab as possible collaborators

Minimal Xen

Create minimal Kbuild for Xen as a reference, using Renesas R-Car as starting point (WIP)

Lead Community Members

- Stefano Stabellini
- EPAM, Dornerworks, XILINX and others as collaborators

Reliable data about achievable minimal code size and community challenges that need to be resolved

Note: *Dom0 and Minimal Xen do not need to be complete to get sufficient data*

Certification Partners

1 WIP: Identify possible certification partners and understand the framework they are willing to work with.

Note: *Dornerworks is a possible partner given past certification experience on Xen*

2 Formalize relationship between vendor(s) and the project

Stage 2:

Create **shared** certification artefacts under the guidance/with support from certification partner
Adapt development processes, where feasible.

Enabling system certification

FreeRTOS based Dom0

- **Lead: EPAM**
- Commercial safety certified version called SafeRTOS
- Baseline for a Renesas FreeRTOS build: 9 SKLOC
- Would need Xenbus, Xenstore and a Toolstack (most functionality in XL is not needed)
- Hardware domains (or other similar disaggregation techniques) would be needed for drivers
- Working first prototype expected in Q3’2018

Dom0-less option

- **Lead: Stefano (Xilinx) & Praveen Kumar (Washington University)**
- Much less clear on how this may work, how much work it is and whether such an approach fulfils safety requirements
- Could build on “Boot multiple VMs in parallel from Device Tree” ➜ clearly has disadvantages for x86
- Demonstrate that after boot Dom0 cannot affect the system anymore

ViryaOS

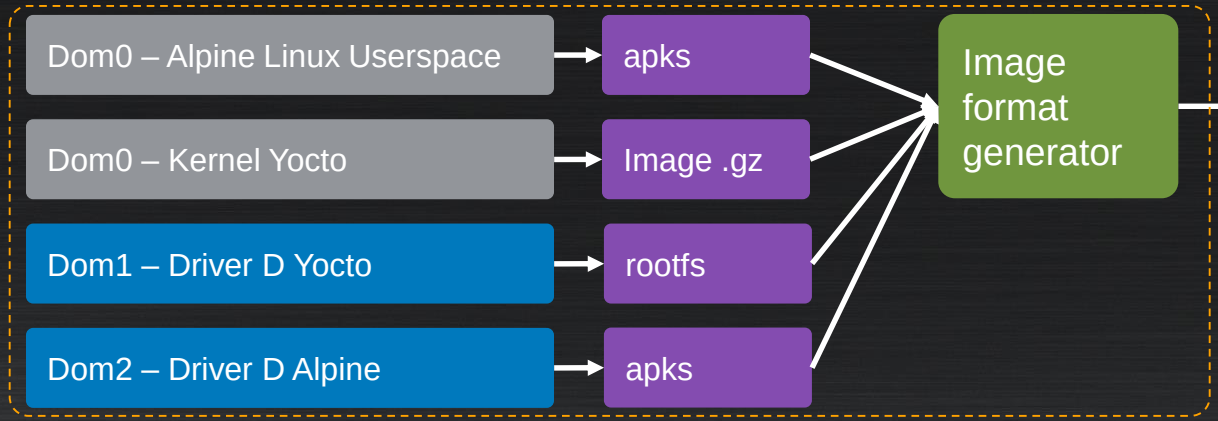
A new proposal for a sub-project: development work has started around 6 months ago to have something concrete when launching

- A **Flexible** build system
- A **Secure** Xen based runtime
- **Containers** supported natively
- Supports ARM and x86
- Targeted at Embedded and IoT

Proposal open for discussion @ lists.xenproject.org/archives/html/xen-devel/2018-05/msg01097.html

ViryaOS: Build

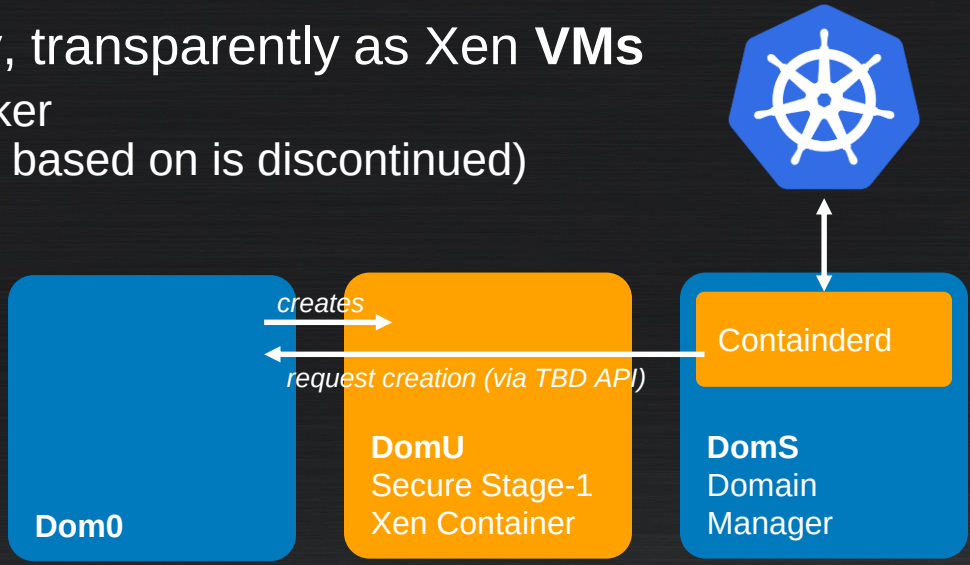
- A **multi-domain** build system
- Builds multiple domains in one go
- Creates a runnable SD Card image from multiple domain builds
- Each domain build is independent and runs in a container
- Pre-configured Device Assignments



Build processes and interimbuild images are docker containers
➜ ease of integration for 3rd party build systems

ViryaOS: Runtime

- Uses **disaggregation**, service domains and driver domains
- Measured Boot
- Supports VMs and containers
- **Containers** are run securely, transparently as Xen **VMs**
  - **Issue:** port stage-1 xen to docker
    (as rkt which stage-1 has been based on is discontinued)



Competing Projects and Efforts in Embedded and Automotive

Competition: Certifiable Hypervisors

L4Re (GPLv2 – dual license) by KernKonzept

- Microvisor with solid feature set
- No public repositories (only code snapshots)
- Requires CLA (Grant of © and patent license)

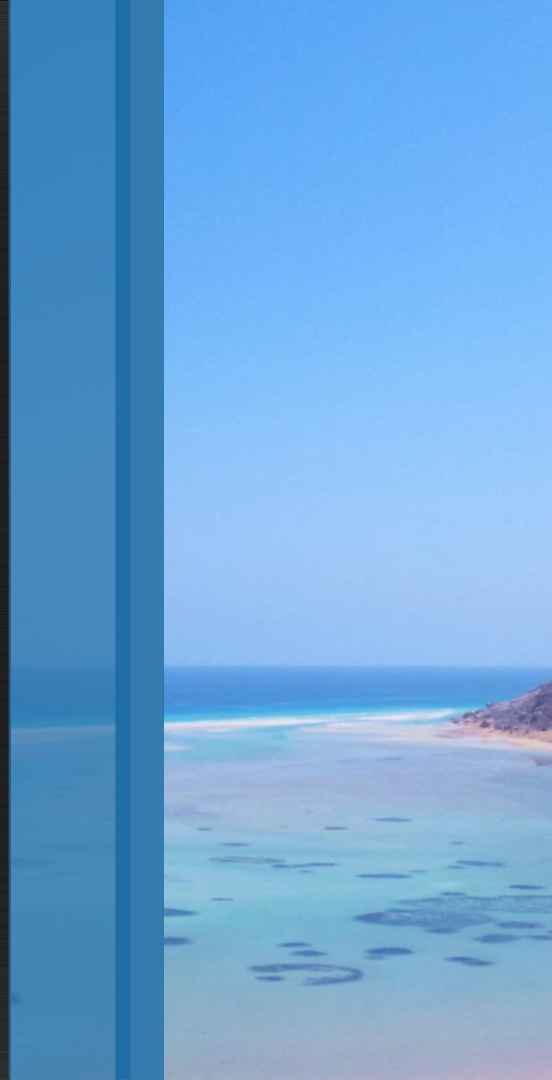
ACRN (3BSD license) by Intel

- x86 only supporting some SKUs; no Arm support; no code separation
- Seems to be mostly a copy of the Xen architecture with some differences (e.g. virtio)
  ➜ Fundamentally a validation of Xen
- Will probably need some time before it becomes mature and complete
- Code size just under 25 SKLOC

Zircon/Fuchsia (BSD/MIT license) by Google

- Not much information available
- PATENTS file prohibits source-code modifications

How You can Help the project



How can you help

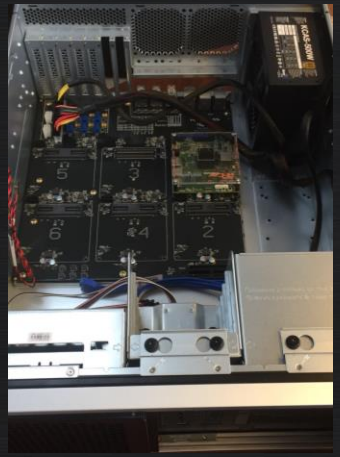
Supporting Development

- Development contributions
- Assign engineers to review design RFCs and patches on upstream mailing list
- Test contributions ➜ in particular for unsupported features
- Documentation contributions ➜ some security functionality is not well documented
- Help guide some of our larger projects / initiatives

How can you help

Funding

- Advisory Board membership
- Sponsoring Events and/or Interns
- Donating Hardware for the Test Lab ➜ e.g. Renesas
- Donate/sponsor Services ➜ e.g. Rackspace
- Travel for engineers to attend community events



Marketing

- User stories, articles, talks at industry events, research …
- The Xen Project can help, but do inform us when you do something


Questions
lars.kurth@xenproject.org
Picture by Lars Kurth