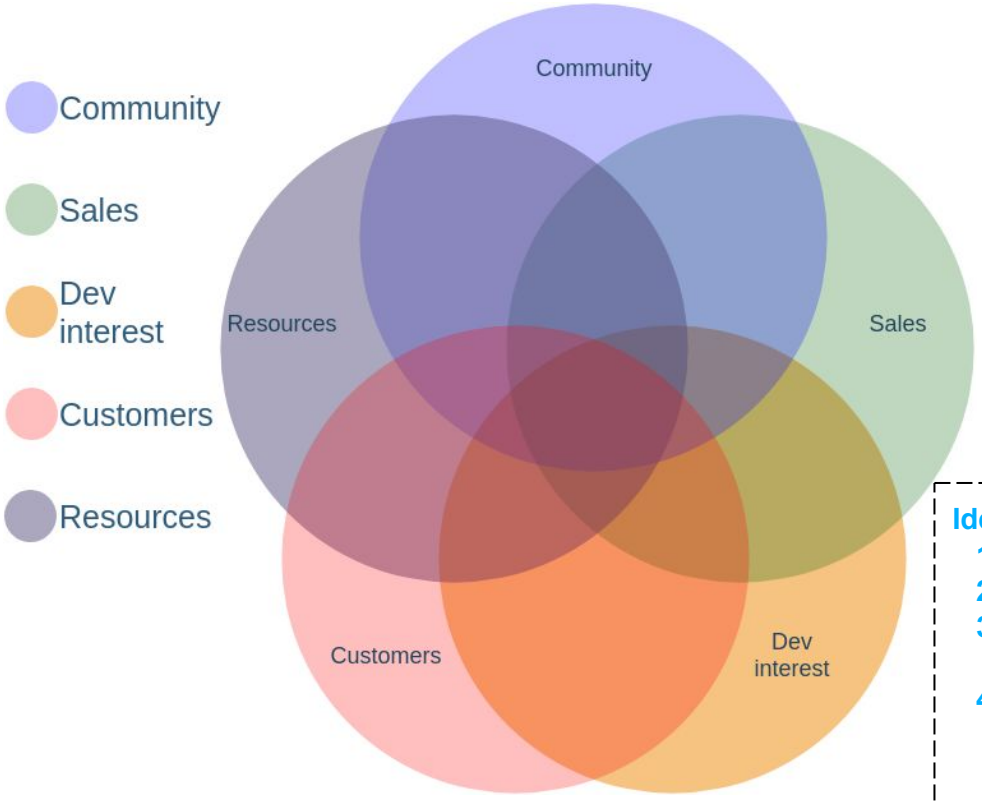
D Scott Phillips
FreeBSD Developer Summit, May, 2019

Intel update & bhyve UEFI

Intel’s BSD Engineering team

Charter: Collaborate with Intel product teams, Intel customers, and larger BSD Community to enable BSD Operating Systems to run optimally on Intel Hardware as well as provide critical support for bugs & security vulnerabilities.

Figuring Out What to Do



Ideally:

1. **community wants**
2. **sales interest**
3. **has a developer available and interested**
4. **customers are ready to deploy.**

hwpstate_intel: hardware-controlled performance

Skylake feature: Processor autonomously selects **performance states**

with consideration of **constraining hints** that are programmed by the OS:

- **minimum** and **maximum** performance limits
- preference towards **energy efficiency** or **performance**
- specification of a relevant workload history observation **time window**.

important MSRs: IA32_HWP_CAPABILITIES, MSR_IA32_HWP_REQUEST, MSR_IA32_HWP_REQUEST_PKG

hwpstate_intel: hardware-controlled performance

Work done by bwidawsk@ https://reviews.freebsd.org/D18028

Get cpufreq(4) to expose per-cpu frequency sysctls.

Need to account for hardware-initiated frequency state changes in cpufreq(4). CPUFREQ_FLAG_UNCACHED

hwpstate_intel: hardware-controlled performance

Performance / Efficiency characteristics:

- generally same-ish as est(4) + powerd(8)
- may improve latency for the first handful of msec from idle
- default settings are a bit more efficiency optimized.

S0ix suspend

Haswell feature: work done by bwidawsk@

Some newer systems do not support the S3 idle state, but can get similar low power usage in S0 when the system is sufficiently idle.

Follow the normal S3 suspend path, then execute idle instruction instead of making ACPI call for S3.

Need to make sure arrangements are made for system wake interrupts (handled by firmware in S3 case).

NVDIMM Namespace support

Similar in purpose to NVME namespaces, splitting an NVDIMM system physical address range up.

Namespaces defined by labels, written into a separate “Label Storage Area” address space in each NVDIMM.

```
typedef struct EFI_NVDIMM_LABEL_SET_COOKIE_INFO {
  typedef struct EFI_NVDIMM_LABEL_SET_COOKIE_MAP {
    UINT64 RegionOffset;
    UINT32 SerialNumber;
    UINT16 VendorId;
    UINT16 ManufacturingDate;
    UINT8 ManufacturingLocation;
    UINT8 Reserved[31];
  } Mapping[NumberOfNvdimmsInInterleaveSet];
};
```

NVDIMM Namespace support

Lots of ACPI tables to let OS know which NVDIMMs are in system physical address range, how interleaved, etc.

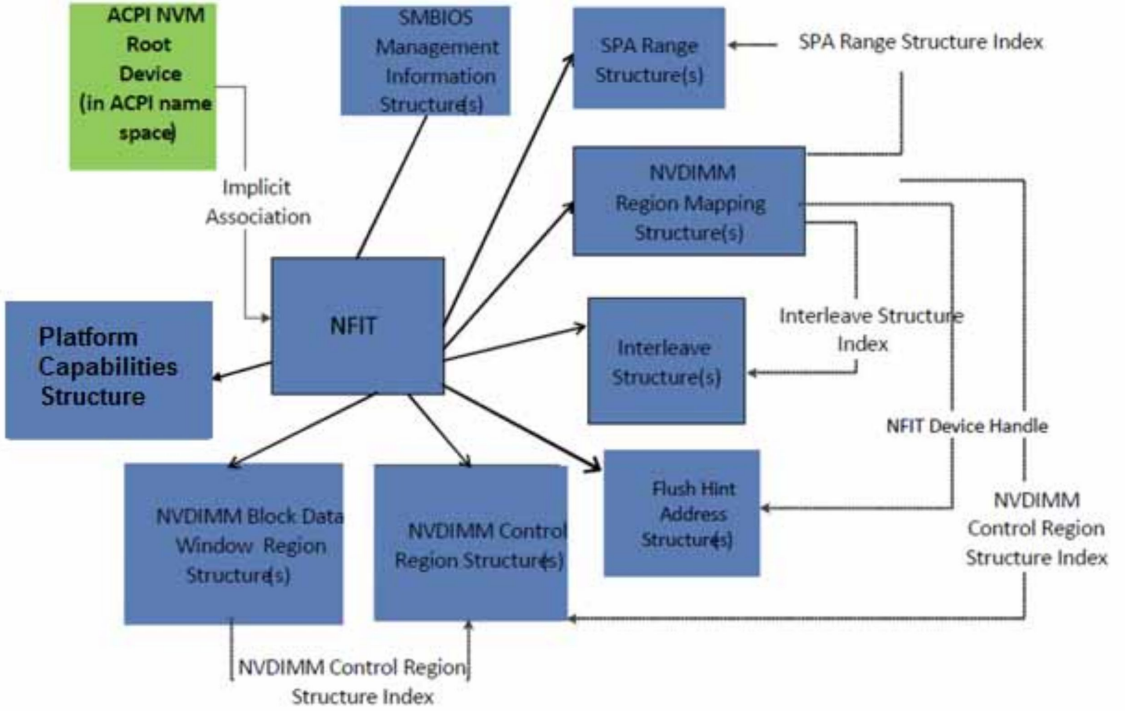


Figure 5-22 NVDIMM Firmware Interface Table (NFIT) Overview

Internal CI system

My current project, make sure pre-production platforms are healthy on CURRENT

Current planned scope is to verify system functionality.

Possible future scope is performance/power. (Inspired by Intel's Linux 0-day testing service).

And now a word from jeb@...

Updating bhyve's UEFI firmware

State of bhyve UEFI

Adaptation of OVMF (from EDK II) for bhyve.

uefi-edk2-bhyve = OVMF - QEMU + bhyve.

(Not the only way to run a VM in bhyve, btw).

Reasons to update

UDK2014.SP1 ← UEFI 2.4B specification, April 2014

edk2-stable201903 ← UEFI 2.7 specification, May 2017

- new ACPI/SMBIOS versions
- Bug fixes/ performance improvements
- HTTP Boot

You can help test it

sysutils/uefi-edk2-bhyve-devel

bug reports welcome

known issue: CSM build currently broken & disabled.

Future updates

https://github.com/tianocore/tianocore.github.io/wiki/EDK-II-Release-Planning

edk2-stable201905 coming soon

- Update OpenSSL version from 1.1.0j to 1.1.1b
- Add new toolchain for LLVM/CLANG8.0
- Replace BSD 2-Clause License with BSD + Patent Licence

More frequent, smaller updates should be less work overall.

Upstreaming? License?

HTTP UEFI Boot

UEFI HTTP Boot

UEFI 2.5 adds HTTP(S) Boot

Network boot without weird configurations or services.

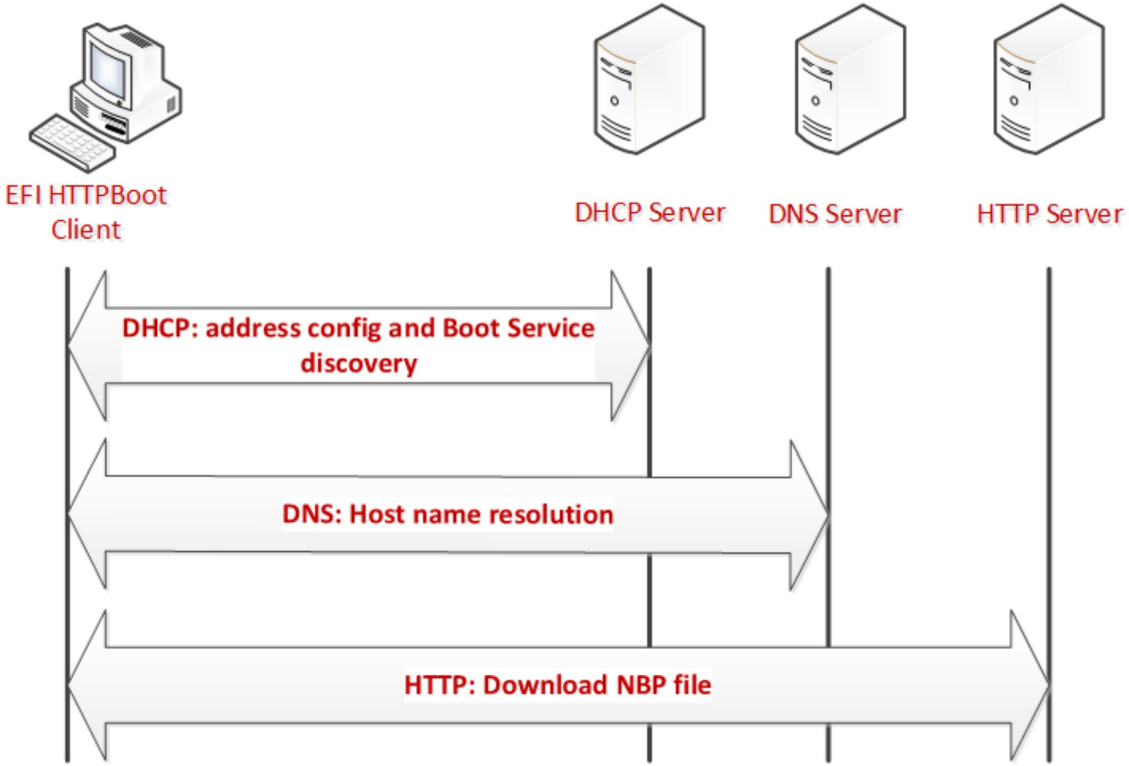


Figure 81. HTTP Boot overall flow

UEFI HTTP Boot

> *the binary image on the boot service is a **UEFI-formatted executable** with a machine subsystem type that corresponds to the UEFI firmware on the client machine, or it could be mounted as a **RAM disk** which contains a UEFI compliant file system*

With a single executable, would want to teach loader.efi to fetch with HTTP.

POC patch to use UEFI's HTTP api at:

`https://gitlab.com/scott-ph/FreeBSD/tree/wip/2019-05/stand-efihttp-poc`

bhyve PXE + NFS root

Convenience script for running bhyve vm with PXE and NFS root from a local directory.

https://gitlab.com/scott-ph/freebsd-dev-vm

https://gitlab.com/scott-ph/freebsd-ports/tree/wip/2019-05/dnsmasq-tftp

INTEL OPEN SOURCE TECHNOLOGY CENTER | 01.org