

Open System Firmware Requirements

Revision 0.3

# 

# Reference to other docs (deeper dive from these principals)

Trammell’s manifesto

* Computer owners must be able to build and install custom versions
* Must be able to redistribute customized versions to other users
* If it breaks you get to keep both pieces

# User driven needs

The following list is derived from discussions of the various vendors over the last months. All these vendors have at least 15 years experience with the UEFI code (and its predecessors) and that experience informs these comments

## Right of redistribution[[1]](#footnote-0)

Binary firmware images built for OCP systems must be freely redistributable, whether included in a system or not, whether modified or not. For example, if a company sells a system to a third party, that third party must be able to resell that system without paying for the firmware. Further, binary images should be available for every OCP system, without royalty, from the OCP web site. There shall be no restriction on an owner’s ability to read firmware from one system and program it on another. Finally, there shall be no limitations imposed on third parties to modify these images and further redistribute them. It must always be possible to reflash an OCP node with new firmware, whether loaded from another node or the web site.

ODMs can redistribute firmware developed for OCP systems for sale to non-OCP customers [bill correct this if it’s wrong].

*Some bit of this text needs to be in the charter. We need indemnification language of some sort. What do we do when a user bricks their box.*

## Binary artifacts relocatable

Binary artifacts must be relocatable to the maximum extent possible. For example, if the CPU requires fixed addresses for some components, e.g. reset vector, then we will not expect to move them. But there should not be gratuitous hard-coded offsets. Any hardcoded addresses must be clearly documented. For example, UEFI implementations should follow the UEFI FV and FFS specifications for locating data and code.

## Open interfaces free from NDA

All interfaces to binary components need to be published and available without requiring an NDA.

## Open Source desired

If source is provided it must be buildable and usable on a board. We do not require the right to redistribute source, although that is highly desirable; we feel that to the maximum extent possible Open Systems Firmware should be built on a foundation of redistributable open source. It is desirable to be able to replace binary blobs with their open source equivalents.

## Blobs based on reproducible builds

If source is not available blobs must be built in a reproducible process.

## Code maintenance

1. We need to integrate with a CI system like Travis CI or similar. Every commit should kick off a build to verify nobody breaks anything. Builds should enable Debug since we have found code that does not build with that enabled.
2. It might be useful to enroll EDK II in coverity

## 

## Toolchain

1. EDK II should build with any widely used C compilers, including GCC and CLANG
2. The build is too slow. 30 minutes needs to become 30 seconds.
3. We would like to have everything work with (at least)GNU make; and to have a system like Kconfig for configuration.

## Build tools

1. Builds need to be reproducible. There seems to be lots of date stamps, paths, and so on that make reproducible builds impossible.
2. Have to be able to scan FVs and modify in situ. Resizing should not require a build cycle. This must be scripted, not an interactive tool.
3. DXEs: need a way to quickly determine, from the command line, given a DXE, what protocol GUIDs it publishes and consumes.

## Resume here next 3/1/2018

## Automated test

1. We need a way to automate unit tests on real hardware, including an ability to recover from bricking a node.
   1. Able to brick the node and recover via out of band mechanism
   2. Possible remote locations for various targets
   3. Software only updates should be possible (via flashrom or flashtool)

## Bug reporting

1. Document the procedure for reporting bugs.
2. Leverage existing GitHub bug tracking mechanism to log and track the bugs related to OSF

## Issues related to booting (both latency and functional)

1. There should never be a prompt required for proper boot. We need a way to boot straight through with zero interaction required.
2. “Boots not bricks”. There are still too many cases in which UEFI will brick a node when things go wrong. We need to fix this.
3. “Boots not bricks” even applies if the firmware fails signature test or other measurement. In other words a security failure should not brick the node, i.e., OCP should not design a system that bricks if security fails. The actual boot policy should be left to the administrator.
4. Platform lock down should be left to the runtime; FLOCKDN/BIOS\_CNTL/TSEG should not be set in PEI.

## Telemetry and diagnosability

1. Strings not error codes during both build and runtime
2. Error logging - It is desirable for firmware to log errors isuch that it can be retrieved out-of-band (e.g. catastrophic failure) or by reading the flash from the OS.
3. Boot logging - It is desirable for firmware to store a boot log (e.g. what it prints to the debug console) such that it can be read from the OS.

## Firmware Variables

1. Need a set of EFI variables with rw access that doesn’t exclusively require efi runtime services, e.g. they should reside in a FV or other container format.
2. EFI variables to control the boot path in PEI phase need to provide ability to be controlled by different DXE paths.

## General platform issues

1. OSF architecture should support ability to “hand over the platform relatively unlocked. No smm, prr, flockdn, bioscntl, tseg, etc. “ so that specific OS interfaces can be developed to install SMM drivers and configure platforms protection bits.
2. Tools like chipsec will be used to verify that in the default configuration the platform protection bits are set to a safe state.

## Vendor-specific issues

These vendor-specific issues are a big pain point for some of our developers and need to be addressed. A common requirement is that all builds must be reproducible and, as mentioned, OS- and compile-agnostic.

### Intel

1. FSP with redistribution clause (Vincent said he was working on this)
2. Source code for Startup ACM (sort of like tboot -- even if we can't sign it we can verify it)
3. Bootguard documentation (hinted at in TXT docs)
4. Remove the reverse engineering clause on the ACM licenses.

### AMD

1. AGESA source or binary with redistribution clause
2. Modules used by PSP must be verifiable.

### ARM

1. SoC vendor must provide silicon init modules either as source or as binary with redistribution clause.
2. ATF modules must be upstreamed.
3. If a TEE is shipped on an OCP node, it should be open (such as OP-TEE).

### POWER

1. Anything needed? It’s already very good as far as firmware goes.

1. This text may seem excessive but every single sentence represents an experience with a vendor trying to limit owners’ ability to reflash firmware. [↑](#footnote-ref-0)