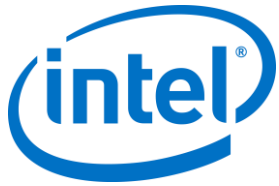

Intel® Scalable System Framework Architecture Specification

May 27, 2016
Version 2016.0

Chapter 0. Introduction

0.1. Overview

Intel® Scalable System Framework (Intel® SSF) is a collection of system building blocks, known as "Intel® SSF elements", and supporting reference architectures. Intel® SSF elements span the domains of compute, networking, storage, and software. They are designed to work together and in concert with other system components to ease the creation of high-performance systems. Each Intel® SSF reference architecture (RA) specifies a high-level system architecture and application platform by documenting industry best practices and conventions. RAs serve to foster innovation and differentiation in system design while preserving application compatibility. Systems using Intel® SSF configurations implement one or more reference architectures.

The specification consists of a modular set of requirements, grouped by chapter and section. An implementation may choose the sections to which it is conformant, subject to dependencies between sections. However, an implementation claiming conformance to a section must satisfy all the requirements of the section. In cases where requirements overlap between sections, the implementation must conform to the most restrictive requirement. Each section has a corresponding identifier that must be defined by the implementation to indicate conformance with the section requirements (see section 1.1 for details).

In its initial version, the specification targets classic HPC clusters, which support traditional modeling and simulation workloads, including typical machine learning workloads. Over time, Intel® SSF will include a family of reference architectures that span small systems through supercomputers and support cloud, data analytics, machine learning, and visualization workloads in addition to traditional HPC workloads. Chapter 1 contains the core requirements common to all reference architectures, while section 2.1 contains requirements specific to a classic HPC cluster. Section 3.1 describes baseline requirements for application compatibility spanning multiple workload domains, while section 3.2 describes requirements for application compatibility of typical HPC workloads.

Beyond the core and reference-architecture-specific requirements, additional sections in Chapter 3 and Chapter 4 define requirements for solution building block elements. These sections may specify capabilities, such as a high-performance messaging system, or components, such as Intel® Omni-Path Architecture. Implementations that comply with a reference architecture's requirements can be enhanced with compliance to additional capability or component sections. Sections describing these requirements will be forthcoming and may be appended to this document. Table 1 summarizes the mapping between sections and reference architectures.

Table 1: Map of SSF_VERSION section identifiers to reference architecture. Sections marked '√' are required, sections marked '◊' are recommended, and empty cells indicate an optional section. Please refer to each section for dependencies and other details.

SECTION IDENTIFIER	SECTION NUMBER	CLASSIC HPC CLUSTER	<FUTURE REFERENCE ARCHITECTURE>
core-2016.0	1.1	√	√
hpc-cluster-2016.0	2.1	√	TBD
compat-base-2016.0	3.1	√	TBD
compat-hpc-2016.0	3.2	√	

0.1.1. Conventions

The key words/phrases "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119[1].

Text with a grey background is advisory.

This is advisory text.

0.1.2. Terminology

COTS

Commercial off-the-shelf (COTS) components are standard manufactured products, not custom products.

Compute node

Compute nodes are the primary computational resource of a system.

Distribution or system software distribution

A system software distribution comprises an operating system kernel, user-space tools and libraries, as well as the documentation. The components of the distribution may originate from different sources, but they are packaged together and released by the distribution supplier; the distribution supplier may be a commercial entity or community based.

External node

An external node provides resources to the system but is not managed as part of the system.

Fully-qualified path name

A fully-qualified path name is the full path of a directory or file, including the root directory in the virtual filesystem hierarchy.

[1] RFC 2119: https://www.ietf.org/rfc/rfc2119.txt

Head node

Some systems combine the logical functions of login, management, and/or service nodes into a single physical system known as the head node.

Job

A user workload running on one or more allocated compute nodes is known as a job. Depending on the system configuration, a job may be launched as a script submitted to the resource manager / scheduler or may be launched directly from the login node.

Login node

The login node is typically the primary point for user interaction with the system. Users initiate jobs from this node, either by submitting jobs to the resource manager or by directly starting job execution. The login node may be shared with multiple users and serves as an interactive resource to stage input and output files, monitor workloads, and build software.

Management node

Management nodes typically support low-level system management functionality and may be very limited in capability.

Node

An independent computer connected to the management network. It has a processor, memory, and at least one network connection with a distinct MAC address.

Service node

A service node provides non-computational resources to the system in support of a job. Users do not typically access a service node directly, but instead through the service that the node supplies.

User-accessible node

Any node where a user can directly start processes. User access may be temporarily granted in cases where a resource manager is used to schedule user workloads.

Chapter 1. Core

1.1. Core

1.1.1. Configuration Information and Compliance

A. Every node shall provide a text file with the pathname "/etc/ssf-release" containing the `SSF_VERSION` field using POSIX* shell variable assignment syntax. The value of the field shall be a colon separated list of section identifiers indicating the sections to which the implementation conforms.

Advisory

The value of SSF_VERSION contained in the /etc/ssf-release file provides the mechanism for a system to list compliance to specific versions of each section. SSF_VERSION should be updated accordingly to reflect changes in compliance as a system is modified over time.

B. The `SSF_VERSION` identifier for this section is `core-2016.0`.

1.1.2. Hardware

A. Each compute node shall have at least one Intel® 64 architecture processor.

Advisory

This section contains the minimum hardware requirements. Additional hardware may be necessary for a fully functional node.

1.1.3. Operating System and Kernel

A. The compute node kernel shall be based on Linux* kernel version 2.6.32 or later.

B. Login and service nodes shall be based on an enterprise Linux distribution with Linux* kernel version 2.6.32 or later.

Advisory

Any Linux* distribution kernel based on this version or later satisfies this requirement.

Enterprise Linux distributions that satisfy this requirement include Red Hat Enterprise Linux*, SUSE Linux Enterprise Server*, Ubuntu LTS*, and derived variants such as CentOS* and Scientific Linux*.

The compute node kernel should be based on Linux* kernel version 3.10.0 or later.

1.1.4. Programming Interfaces

A. All required APIs shall be defined using the LP64 programming model on all nodes.

Advisory

Where explicitly required for compatibility with existing practice or current program binaries, other programming model APIs may be provided in addition to LP64.

B. The ABI behavior shall conform to the Intel® 64 architecture on all nodes.

Advisory

The Intel® 64 architecture ABI is also known as 'x86_64'.

Where explicitly required for compatibility with existing practice or current program binaries, other ABIs may be provided in addition to Intel® 64 architecture.

1.1.5. Runtime Environment

A. At login, the environment variable `$HOME` shall be set on all nodes to the fully-qualified pathname of the user's home directory.

B. The environment variable `$TMPDIR` shall be set on all nodes to the fully-qualified pathname of a node-private temporary directory. If the temporary directory is contained within a shared filesystem, then the value of `$TMPDIR` shall be unique to each node.

Advisory

The temporary directory is not required to be local or persistent.

On user accessible nodes, the environment variable `$SCRATCH` should be set to the fully-qualified pathname of a persistent directory on a high performance filesystem to host temporary user workload files.

C. Environment modules shall be used to allow multiple runtime environments to coexist on user accessible nodes.

Advisory

This includes multiple versions of the same component.

It is left to the implementer to choose which runtime environments are default, if any.

The module naming convention should use the name and version of the component, e.g., `impi/5.1.1.109`. Where a version is not applicable, the version may be omitted. A hierarchical organizational scheme should be used to handle dependencies, e.g., on a compiler family.

Chapter 2.
Base Reference Architectures

2.1. Classic High Performance Compute Cluster

The “classic” High Performance Compute cluster typically uses commercial off-the-shelf components to form a parallel computing platform. Applications that target this type of system typically use the message passing interface (MPI) for parallel execution.

This reference architecture typically employs a head node that serves to manage the system, be the primary login/interface for users, and provide numerous system wide functions and utilities. Compute nodes based on Intel® 64 architecture processors are the primary computational resource. A user workload running on one or more allocated compute nodes is known as a job. Additional nodes may provide specific, non-compute cluster services, such as login nodes, storage nodes, etc. Typically only the compute and login node(s) are directly accessible by users.

These clusters have at least three distinct communications needs: application messaging, system management, and cluster-wide storage. From an architectural standpoint, each of these networks has distinct logical requirements. However, implementations may choose to combine one or more of these logical networks in a single physical fabric.

2.1.1. Configuration Information and Compliance

A. The `SSF_VERSION` identifier for this section is `hpc-cluster-2016.0`.

B. If an implementation claims compliance to this section, then `SSF_VERSION` must also contain the `core-2016.0` section identifier and meet all corresponding requirements.

2.1.2. Operating System and Kernel

Advisory

For the general case, the hardware components and software environment of compute nodes should be uniform. Files that specify unique identification or configuration of the node may be different as needed.

In cases where specialized nodes are desired, a resource manager should be provided and should comprehend any differences.

2.1.3. Network Fabrics

A. Each compute node’s network host name shall be consistently resolved to its network address.

B. The management fabric shall be accessible using the standard IP network stack.

C. All login nodes shall be externally accessible via SSH.

2.1.4. Authentication and Access Control

A. All nodes shall operate under a single authentication domain, *i.e.,* once authenticated, one set of credentials shall permit access to the cluster.

Advisory

User access may be limited to a subset of the system.

B. Privileged users shall be able to execute commands on all nodes.

C. Unprivileged users shall be able to execute commands on all currently allocated compute nodes.

Advisory

Unprivileged users are not required to be able to access a compute node unless they have been allocated resources on that node.

Unprivileged users are not required to have interactive access to a compute node even if they have been allocated resources on that node.

D. Unprivileged users shall be able to access their data stored locally on currently allocated compute nodes.

Advisory

Unprivileged users are not required to be able to access data stored locally on a compute node unless they have been allocated resources on that node.

Chapter 3. Capabilities

3.1. Base Application Compatibility Requirements

An ecosystem of highly compatible systems provides a consistent application target for application developers. Conversely, conforming systems enjoy a wealth of compatible applications. Solutions that comply with this section present a known interface to the application layer. In turn, application developers and vendors can compile and distribute binaries for this target platform, enabling application binary mobility.

This section enumerates the baseline interfaces and minimum hardware requirements necessary for application compatibility across a range of reference architectures.

3.1.1. Configuration Information and Compliance

A. The `SSF_VERSION` identifier for this section is `compat-base-2016.0`.

B. If an implementation claims compliance to this section, then `SSF_VERSION` must also contain the `core-2016.0` section identifier and meet all corresponding requirements.

3.1.2. Hardware

Advisory

Minimal hardware requirements are described to ensure functional systems are built from this platform definition. This specification does not guarantee that specific implementations built only to these minimal requirements will provide optimal application performance. Implementers must assume that additional hardware resources beyond this set of minimums may be required to provide optimal application performance.

A. Each compute node shall have

   a. a minimum of 16 gibibytes of random access memory;

   b. access to at least 80 gibibytes of persistent storage.

      Advisory

      The storage may be implemented as direct access local storage or available over a network.

      The storage may be globally visible or node private.

B. Login nodes shall have at least 200 gibibytes of persistent storage.

   Advisory

   The storage may be implemented as direct access local storage or available over a network.

3.1.3. Operating System and Kernel

Advisory

The operating system on user accessible nodes should materially conform to the Linux* Standard Base 5.0 core specification[2].

[2]Linux Standard Base 5.0: http://refspecs.linuxfoundation.org/LSB_5.0.0/allspecs.shtml

3.1.4. Programming Interfaces

Advisory

Some applications may require additional or newer versions of these runtimes for compatibility.

A. A materially conformant POSIX.1-2008 API[3] shall be provided on user accessible nodes.

B. The following Linux Standard Base* (LSB) 5.0 runtime libraries[4] shall be provided on user accessible nodes and recognized by the dynamic loader for the Intel® 64 architecture:

LIBRARY	RUNTIME NAME
libc	libc.so.6
libcrypt	libcrypt.so.1
libdl	libdl.so.2
libgcc_s	libgcc_s.so.1
libm	libm.so.6
libncurses	libncurses.so.5
libncursesw	libncursesw.so.5
libpam	libpam.so.0
libpthread	libpthread.so.0
librt	librt.so.1
libstdcxx	libstdc++.so.5, libstdc++.so.6
libutil	libutil.so.1
libz	libz.so.1
proginterp	/lib64/ld-linux-x86-64.so.2, /lib64/ld-lsb-x86-64.so.3

Advisory

Where explicitly required for compatibility with existing practice or current program binaries, other runtime libraries and ABIs may be provided in addition to Intel® 64 architecture.

The following additional runtime libraries should be provided on all user accessible nodes and recognized by the dynamic loader for the Intel® 64 architecture:

- libBrokenLocale.so.1
- libSegFault.so
- libanl.so.1
- libacl.so.1
- libattr.so.1
- libbz2.so.1
- libcap.so.2
- libcrypto.so.6
- libnsl.so.1
- libnss_compat.so.2
- libnss_dns.so.2

[3] POSIX.1-2008: http://pubs.opengroup.org/onlinepubs/9699919799/

[4] Linux Standard Base Core 5.0 Core Specification: http://refspecs.linuxfoundation.org/LSB_5.0.0/LSB-Core-generic/LSB-Core-generic/requirements.html, http://refspecs.linuxfoundation.org/LSB_5.0.0/LSB-Core-AMD64/LSB-Core-AMD64/requirements.html
Linux Standard Base 5.0 Desktop Specification: http://refspecs.linuxfoundation.org/LSB_5.0.0/LSB-Desktop-generic/LSB-Desktop-generic/requirements.html

- libnss_files.so.2
- libnss_hesiod.so.2
- libnss_ldap.so.2
- libnss_nis.so.2
- libnuma.so.1
- libpanel.so.5
- libpanelw.so.5
- libresolv.so.2
- libthread_db.so.1

C. The LP64 version of the following runtime libraries shall be provided on user accessible nodes and with runtime environments configurable using environment modules:

a. ANSI* standard C/C++ language runtime of the GNU* C Compiler version 4.8 or later

b. ANSI* standard C/C++ language runtime of the Intel® C++ Compiler version 16.0 or later

c. Intel® Math Kernel Library version 11.3 or later

d. Intel® Threading Building Blocks version 4.4 or later

Advisory

The corresponding environment modules should be loaded by default.

For each component, the runtimes are defined to include all of the runtime libraries distributed with the component. E.g., the OpenMP* runtime library is included as part of the Intel® C++ Compiler runtime.

The identified Intel runtime components above are provided without fee.

3.1.5. Runtime Environment

A. Users' home directories shall reside on a shared file system that is mounted on all user accessible nodes.

3.1.6. Command System and Tools

A. The following subset of the Linux Standard Base* (LSB) 5.0 command system[5] shall be provided on all user accessible nodes:

[5] Linux Standard Base 5.0 Core Specification: http://refspecs.linuxfoundation.org/LSB_5.0.0/LSB-Core-generic/LSB-Core-generic/command.html

[	cut	find	logname	pathchk	strings
ar	date	fold	ls	pidof	tail
awk	dd	fuser	mkdir	printf	tar
basename	diff	getconf	mkfifo	ps	tee
bc	dirname	grep	mktemp	pwd	test
cat	du	head	more	rm	time
chmod	echo	hostname	mv	rmdir	touch
chown	egrep	iconv	nice	sed	tr
cksum	env	id	nl	seq	true
cmp	expr	join	nohup	sh	uname
comm	false	kill	od	sleep	uniq
cp	fgrep	killall	paste	sort	wc
csplit	file	ln	patch	split	xargs

Advisory

A complete and materially conformant POSIX.1-2008* command system[6] should be provided on all user accessible nodes.

A complete and materially conformant Linux Standard Base* (LSB) 5.0 command system[5] should be provided on all user accessible nodes.

B. The Python* scripting language version 2.6.6 or later shall be provided on user accessible nodes.

Advisory

Python* scripting language version 3.0 or later should be provided on user accessible nodes, in addition to version 2.6.6 or later.

C. The Perl* scripting language version 5.10 or later shall be provided on user accessible nodes.

D. The Tcl scripting language version 8.5 or later shall be provided on user accessible nodes.

3.2. High Performance Computer Cluster Application Compatibility Requirements

An ecosystem of highly compatible "classic" HPC clusters provides a consistent application target for application developers. Solutions that comply with this section present a known interface to the application layer. In turn, application developers and vendors can compile and distribute binaries for this target platform, enabling application binary mobility.

While this section is intended primarily for MPI applications distributed as binaries, it is also applicable for MPI applications built from source on the intended system as well as non-MPI workloads.

3.2.1. Configuration Information and Compliance

A. The `SSF_VERSION` identifier for this section is `compat-hpc-2016.0`.

[6] POSIX.1-2008 Utilities: http://pubs.opengroup.org/onlinepubs/9699919799/idx/utilities.html

B. If an implementation claims compliance to this section, then `SSF_VERSION` must also contain the `hpc-cluster-2016.0` and `compat-base-2016.0` section identifiers and meet all corresponding requirements.

3.2.2. Hardware

Advisory

Minimal hardware requirements are described to ensure functional systems are built from this platform definition. This specification does not guarantee that specific implementations built only to these minimal requirements will provide optimal application performance. Implementers must assume that additional hardware resources beyond this set of minimums may be required to provide optimal application performance.

A. Each compute node shall have a minimum of 32 gibibytes of random access memory.

3.2.3. Programming Interfaces

Advisory

Some applications may require additional or newer versions of these runtimes for compatibility.

A. The following Linux Standard Base* (LSB) 5.0 runtime libraries[7] shall be provided on user accessible nodes and recognized by the dynamic loader for the Intel® 64 architecture:

LIBRARY	RUNTIME NAME
libGL	libGL.so.1
libGLU	libGLU.so.1
libICE	libICE.so.6
libSM	libSM.so.6
libX11	libX11.so.6
libXext	libXext.so.6
libXft	libXft.so.2
libXi	libXi.so.6
libXrender	libXrender.so.1
libXt	libXt.so.6
libXtst	libXtst.so.6
libfontconfig	libfontconfig.so.1
libfreetype	libfreetype.so.6
libjpeg	libjpeg.so.62
libxcb	libxcb.so.1

Advisory

Where explicitly required for compatibility with existing practice or current program binaries, other runtime libraries and ABIs may be provided in addition to Intel® 64 architecture.

The following additional runtime libraries should be provided on all user accessible nodes and recognized by the dynamic loader for the Intel® 64 architecture:

- `libXau.so.6`
- `libXcursor.so.1`

[7] Linux Standard Base 5.0 Desktop Specification: http://refspecs.linuxfoundation.org/LSB_5.0.0/LSB-Desktop-generic/LSB-Desktop-generic/requirements.html

- libXfixes.so.3
- libXinerama.so.1
- libXmu.so.6
- libXp.so.6
- libXrandr.so.2
- libXxf86vm.so.1
- libpng12.so.0

B. The LP64 version of the following runtime libraries shall be provided on user accessible nodes and with runtime environments configurable using environment modules:

   a. Standard Fortran language runtime of the Intel® Fortran Compiler version 16.0 or later

   b. Intel® MPI Library Runtime Environment version 5.1 or later

Advisory

The corresponding environment modules should be loaded by default.

For each component, the runtimes are defined to include all of the runtime libraries distributed with the component. E.g., the OpenMP* runtime library is included as part of the Intel® Fortran Compiler runtime.

The identified Intel runtime components above are provided without fee.

3.2.4. Storage and File System

A. Cluster file systems shall provide at least the consistency and access guarantees provided by NFS version 3.0[8].

Advisory

Cluster file systems should provide at least the consistency and access guarantees provided by NFS version 4.0[9].

Cluster file systems should support POSIX.1-2008 file semantics[10].

[8] Network File System (NFS) version 3 Protocol (RFC 1813): https://www.ietf.org/rfc/rfc1813.txt

[9] Network File System (NFS) version 4 Protocol (RFC 3530): https://www.ietf.org/rfc/rfc3530.txt

[10] POSIX.1-2008 System Interfaces: http://pubs.opengroup.org/onlinepubs/9699919799/functions/V2_chap02.html

Chapter 4. Components

This chapter is a placeholder for future content.