

Intel® VTune™ Amplifier 2019 Installation Guide - Linux* OS

Intel Corporation
www.intel.com
Legal Information

Contents

Legal Information 4

Chapter 1: Introduction

Chapter 2: Prerequisites

Chapter 3: Installation Steps

Installing with the Intel® VTune™ Amplifier Installer Graphical User Interface 8
Installing Intel® VTune™ Amplifier with the Command Line Installer 9
Automated Installation of Intel® VTune™ Amplifier 9
Installing Intel® VTune™ Amplifier in a Cluster Environment 10
Installing Intel® VTune™ Amplifier for Use with a Virtual Machine 13

Chapter 4: Post-Installation Steps

Set Up Linux* System for Remote Analysis 16
Enable Linux* Kernel Analysis 18
Embedded Linux* Targets 19
Configure Yocto Project* and VTune™ Amplifier with the VTune Amplifier Integration Layer 25
Configure Yocto Project*/Wind River* Linux* and Intel® VTune™ Amplifier with the Intel System Studio Integration Layer 26
Configure Yocto Project* and Intel® VTune™ Amplifier with the Linux* Target Package 28
Set Up FreeBSD* System 29
Set Up Android* System 30
Enable Java* Analysis on Android* System 32
Prepare an Android* Application for Analysis 35
Search Directories for Android* Targets 36
Targets in Virtualized Environments 36
Profile Targets on a VMware* Guest System 38
Profile Targets on a KVM* Guest System 39
Profile KVM Kernel Modules from the Host 41
Profile KVM Kernel and User Space on the KVM System 43
Profile KVM Kernel and User Space from the Host 46
Profile Targets on a Xen* Virtualization Platform 47
Targets in a Cloud Environment 48
Configure SSH Access for Remote Collection 49
Configure system for GPU Application Analysis on Intel® HD Graphics and Intel® Iris® Graphics 49
Rebuild and Install the Kernel for GPU Analysis 50
Verify Intel® VTune™ Amplifier Installation on a Linux* System 51
Installing Command Line Collectors 51
Sampling Drivers 52
Build and Install the Sampling Drivers for Linux* Targets 52
Build and Install Sampling Drivers for Android* Targets 56
Driverless Event-Based Sampling Collection 57
Platform Profiler Setup (Preview) 59

Chapter 5: Getting Started with Intel® VTune™ Amplifier

Chapter 6: Intel® VTune™ Amplifier Installation FAQs

How do I install the drivers required by Intel VTune Amplifier?..... 62
Error Message: Installing and Starting VTSS++ Driver...FAILED..... 62
Error Message: No Pre-built Driver Exists for This System..... 62
What features are available if I install Intel VTune Amplifier as a non-root or non-administrator user? 63
How do I configure remote data collection using Intel VTune Amplifier?..... 63
Can I install multiple versions of Intel VTune Amplifier on the same system? ... 64
Problem: Platform Not Authorized..... 64
Error Message: Cannot Collect GPU Hardware Metrics..... 64

Chapter 7: Uninstalling Intel® VTune™ Amplifier

Chapter 8: Additional Resources

Legal Information

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors which may cause deviations from published specifications. Current characterized errata are available on request.

Intel, the Intel logo, Intel Atom, Intel Core, Intel Xeon Phi, VTune and Xeon are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

Intel, the Intel logo, Intel Atom, Intel Core, Intel Xeon Phi, VTune and Xeon are trademarks of Intel Corporation in the U.S. and/or other countries.

Microsoft, Windows, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Java is a registered trademark of Oracle and/or its affiliates.

OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission by Khronos.

Copyright 2015-2019 Intel Corporation.

This software and the related documents are Intel copyrighted materials, and your use of them is governed by the express license under which they were provided to you (**License**). Unless the License provides otherwise, you may not use, modify, copy, publish, distribute, disclose or transmit this software or the related documents without Intel's prior written permission.

This software and the related documents are provided as is, with no express or implied warranties, other than those that are expressly stated in the License.

1 Introduction

Use VTune Amplifier to locate or determine:

- The most time-consuming (hot) functions in your application and/or on the whole system
- Sections of code that do not effectively utilize available processor time
- The best sections of code to optimize for sequential performance and for threaded performance
- Synchronization objects that affect the application performance
- Whether, where, and why your application spends time on input/output operations
- The performance impact of different synchronization methods, different numbers of threads, or different algorithms
- Thread activity and transitions
- Hardware-related issues in your code such as data sharing, cache misses, branch misprediction, and others

This document explains how to install and configure Intel VTune Amplifier on a Linux* system.

To install VTune Amplifier, you can use:

- Graphical user interface (GUI) installer: Presents installation options and allows you to choose product components.
- Command line installer: Uses a command prompt to present installation options and allow you to choose product components.
- Non-interactive installer: Uses a single command to install the product in the background.

NOTE
A 64-bit operating system host is required to use the VTune Amplifier graphical user interface to analyze collected profile data. Command line profiling and reporting is supported on a 32-bit operating system host. Use the command line installer instructions to install on a 32-bit operating system.

2 Prerequisites

The following information is important to consider before beginning to install Intel® VTune™ Amplifier.

System Requirements

Review the full list of system requirements listed in the Release Notes document. The document can be found online and in your installation media.

A 64-bit operating system host is required to use the VTune Amplifier graphical user interface to analyze collected profile data. Command line profiling and reporting is supported on a 32-bit operating system host. Use the command line installer instructions to install on a 32-bit operating system.

User Permissions for Install

The VTune Amplifier installer is launched using the default user account. The user installing the product should have read and write permissions for the `/tmp` directory.

Most of the Intel VTune Amplifier profiling features work with a non-root install. Many work on either a genuine Intel processor or a compatible processor using a driverless collection. For more information, see Driverless Event-based Sampling Collection.

Some advanced features that use event-based sampling require the latest OS kernel or sampling driver to be installed. Intel Atom® processors also require this driver for analysis. To install the driver, launch the installer as root or ask your system administrator to install the driver later. For information on building and setting up the drivers, see Sampling Drivers.

Required Packages for Intel VTune Amplifier GUI

> **NOTE**
> The Intel VTune Amplifier GUI is not supported for Red Hat* Enterprise Linux/CentOS* 6.

The following packages are required for the Intel VTune Amplifier GUI to be installed:

On Ubuntu* or Debian, install the following libraries:

```
sudo apt-get install libgtk-3-0 libasound2 libxss1 libnss3
```

On Red Hat* Enterprise Linux or CentOS*, install the following libraries:

```
sudo yum install gtk3 alsa-lib libXScrnSaver nss
```

On Fedora*, install the following libraries:

```
sudo dnf install gtk3 alsa-lib libXScrnSaver nss
```

On SUSE* Linux* Enterprise Server (SLES), install the following libraries:

```
sudo zypper install gtk3 libasound2 libXss1 mozilla-nss
```

Previous Versions of Intel VTune Amplifier

You do not need to uninstall previous versions or updates of VTune Amplifier before installing a newer version. However, if you do not remove older updates before installing a newer version, all product components that belong to the same major release will be replaced with the latest version for each major release update.

For example, if you have VTune Amplifier 20xx Update 1 installed and are currently installing VTune Amplifier 20xx Update 2, the older version will be uninstalled and replaced with the new content for Update 2. If you are installing the next major release, VTune Amplifier 20xy, your installation of VTune Amplifier 20xx Update 1 will remain and the new release will be installed beside the old version, duplicating common files, documentation, samples, and product components.

Installing in a Cluster Environment

If you are installing in a cluster environment, you can install and use multiple versions of the product on the same system. However, kernel driver usage is limited to a single version of VTune Amplifier. This means you can have multiple copies of VTune Amplifier installed without the SEP drivers and a single version of the product with the drivers installed. The latter would be enabled with the advanced types of analysis using hardware event based sampling analysis data collection.

3 Installation Steps

The Intel® VTune™ Amplifier installation package contains all components of the product in a downloadable file. The installer can be run as an administrator from a GUI or from a command prompt.

The Intel® Software Manager installs automatically with all Intel® Software Development Products on Windows*, Linux*, and macOS* systems. The Intel Software Manager is a utility that allows users to:

- Download and install updates for your Intel Software Development Products.
- Manage subscription status and activate serial numbers of installed software (if applicable).
- Find out about the latest news for Intel Software Development Products.
- Intel Software Manager requires an internet connection to connect to a remote server for information and updates.

Refer to the following site for more information about Intel Software Manager: https://registrationcenter-ssl.intel.com/Docs/ism.htm

The following sections detail the steps required to install Intel VTune Amplifier.

- Installing with the Installer Graphical User Interface
- Installing with the Installer Command Line
- Installing with the Automated Installer
- Installing in a Cluster Environment
- Installing for Use with a Virtual Machine

Installing with the Intel® VTune™ Amplifier Installer Graphical User Interface

NOTE
Refer to the Prerequisites before you begin the installation.

Use the following steps to launch the installer GUI:

1. Extract the installation package to a writeable directory with the following command:

   ```
   tar -xzf vtune_amplifier_<version>.tar.gz
   ```
2. Navigate to the directory containing the extracted files.
3. Run the following command to launch the installer:

   ```
   ./install_GUI.sh
   ```
4. After installation succeeds, run the following command to establish the VTune Amplifier environment:

   For bash command interpreter:

   ```
   source <install-dir>/amplxe-vars.sh
   ```

   For csh/tcsh command interpreter:

   ```
   source <install-dir>/amplxe-vars.csh
   ```

The installation process includes the following steps:

- Installation Location and Components

  Lists the default installation location and options. Select the components to install and change the default installation location (optional).
- Prerequisites

Lists all prerequisites that would prevent a fully successful installation. Prerequisites could include additional requirements, information about setting up drivers, a reminder to restart your system after installation completes, and so on.

- Options

  Install all options or click **Customize** to select a subset of options.

  Click the **Install** button to begin installation.
- Complete

  The Intel VTune Amplifier Getting Started page displays after installation succeeds.

Installing Intel® VTune™ Amplifier with the Command Line Installer

NOTE
Refer to the Prerequisites before you begin the installation.

Use the following steps to launch the command line installer:

1. Extract the installation package to a writeable directory with the following command:

   ```
   tar -xzf vtune_amplifier_<version>.tar.gz
   ```
2. Navigate to the directory containing the extracted files.
3. Run the following command to run the installer:

   ```
   ./install.sh
   ```

   Tip
   Run the following command to install all components on a network-mounted drive or shared file system: `./install.sh --SHARED_INSTALL`
4. Follow the command prompts to install the product.
5. After installation succeeds, run the following command to establish the VTune Amplifier environment:

   For bash command interpreter:

   ```
   source <install-dir>/amplxe-vars.sh
   ```

   For csh/tcsh command interpreter:

   ```
   source <install-dir>/amplxe-vars.csh
   ```

Automated Installation of Intel® VTune™ Amplifier

NOTE
Refer to the Prerequisites before you begin the installation.

The automated, or silent, installation method allows you to perform a command line installation of Intel® VTune™ Amplifier without answering prompts or making product selections. Use the following steps to set up and execute an automated installation:

1. Copy the working product license to the standard license file directory, such as `/opt/intel/licenses`. If you want to use a non-standard license directory, set the `INTEL_LICENSE_FILE` environment variable to the location of the license file.
2. Download the full installer package.

3. Extract the installation package to a writeable directory with the following command:

   ```
   tar -xzf vtune_amplifier_<version>.tar.gz
   ```
4. Change directories to the location where you extracted the install files.
5. Run the following command to create a configuration file that contains the information required by the installer, where *<filename>* is the name of the new configuration file:

   ```
   ./install.sh -d <filename>.cfg
   ```
6. Update the `*.cfg` file you just created to include the answers to the installer prompts. Save and close the file when you finish making updates.

   The following is an example of the configuration text file:

```
ACCEPT_EULA=accept
CONTINUE_WITH_OPTIONAL_ERROR=yes
PSET_INSTALL_DIR=/opt/intel
CONTINUE_WITH_INSTALLDIR_OVERWRITE=yes
PSET_MODE=install
CLUSTER_INSTALL_AUTOMOUNT=yes
AMPLIFIER_SAMPLING_DRIVER_INSTALL_TYPE=build
AMPLIFIER_DRIVER_ACCESS_GROUP=vtune
AMPLIFIER_DRIVER_PERMISSIONS=666
AMPLIFIER_LOAD_DRIVER=yes
AMPLIFIER_C_COMPILER=/usr/bin/gcc
AMPLIFIER_KERNEL_SRC_DIR=/lib/modules/2.6.32-504.el6.x86_64/build
AMPLIFIER_MAKE_COMMAND=/usr/bin/make
AMPLIFIER_INSTALL_BOOT_SCRIPT=yes
AMPLIFIER_DRIVER_PER_USER_MODE=no
```

7. Run the following command to begin the installation using your configuration file:

   ```
   ./install.sh -s <filename>.cfg
   ```

The product is installed in the directory specified in the configuration file.

Installing Intel® VTune™ Amplifier in a Cluster Environment

There are two typical scenarios for installing Intel® VTune™ Amplifier in a cluster environment:

1. Install and enable the kernel driver on specific nodes and for specific users. The users can launch VTune Amplifier on a node or machine from a shared partition mounted on their system and use this to analyze programs or system behavior on that node or machine. Even though the program's execution may be distributed among other nodes, a single instance of the tool is collecting performance data for the single node on which it is launched. This is the most common use case as it requires less administrative overhead.
2. Install VTune Amplifier on a shared partition that is available to all users. You can either allow the VTune Amplifier installer to automatically install the SEP drivers or you can manually install the drivers after product installation succeeds. This use case requires additional administrative oversight.

There are a few changes to the typical installation steps when installing in a cluster environment. The steps that follow set up a cluster environment so that users that belong to a custom group (`my_group`) can run hardware event based analysis on specific nodes. Users can run VTune Amplifier using the GUI or the command line, depending on their display device.

Tip
VTune Amplifier users in a cluster environment are expected to set the results directory path within their home directory. By default, the tool uses `${HOME}/intel/amplxe/Projects/<project-name>`, but users can set up a directory to save analysis results to a local path. Setting a local path is helpful when network connection speeds are slow as it can help speed data loading and processing while analyzing collected results. For example, a user could specify the `/tmp` directory, but should ensure that enough disk space is available where `/tmp` is mounted.

Advanced Installation

NOTE
Refer to the Prerequisites before you begin the installation.

Use the following steps to complete the installation with more control over driver installation:

1. Run the install script with the following command to skip driver installation on the current machine. Running with the `--SHARED_INSTALL` option is required as users are expected to launch profiling on the compute nodes, in general, not necessarily on the main or head node or the node used by administrator for installation.

   ```
   ./install.sh --SHARED_INSTALL
   ```

   The product installation generates the `amplxe-vars.{sh|csh}` and `amplxe-<version.build_number>` module file for cluster environments that use environment modules. These files are added to the top-level product directory and are used to set up the user environment for launching VTune Amplifier. For more information about environment modules, see http://modules.sourceforge.net.

   NOTE
   Intel VTune Amplifier can still be used for profiling with the predefined analysis types based on software sampling even without the sampling driver installed. Users can launch the product from a shared file system to run the predefined analysis types (hotspots, threading, etc.).

2. Build the SEP driver for the current OS. See the `<install-dir>/sepdk/src/README.txt` document for more details on building the driver.
   - Build in the `src` directory using the following commands:

     ```
     cd <install-dir>/sepdk/src ./build-driver -ni
     ```
   - Build in a custom directory using the `--install-dir` option to specify the installation directory. Make sure that the directories specified in the option already exist.
     1. Run the following command to build and install the driver:

        ```
        ./build-driver --install-dir=
        ```
     2. Use the following commands to copy scripts to the driver installation directory:

        ```
        cp insmod-sep <custom-install-dir> cp rmmod-sep <custom-install-dir> cp boot-script <custom-install-dir>
        ```
     3. Create a pax subdirectory in the new driver location and navigate to the new directory:

        ```
        cd pax
        ```
     4. Use the following commands to copy the pax driver scripts:

```
cp insmod-pax <custom-install-dir>/pax cp rmmod-pax <custom-install-
dir>/pax cp boot-script <custom-install-dir>/pax
```

3. Install the event based sampling kernel driver on the selected nodes.

   Enter the node on which performance profiling will be completed and run the following commands from the shared directory where the driver is located:

   ```
   cd <install-dir>/sepdk/prebuilt
   ./insmod-sep --group my_group
   ./boot-script --install --group my_group
   ```

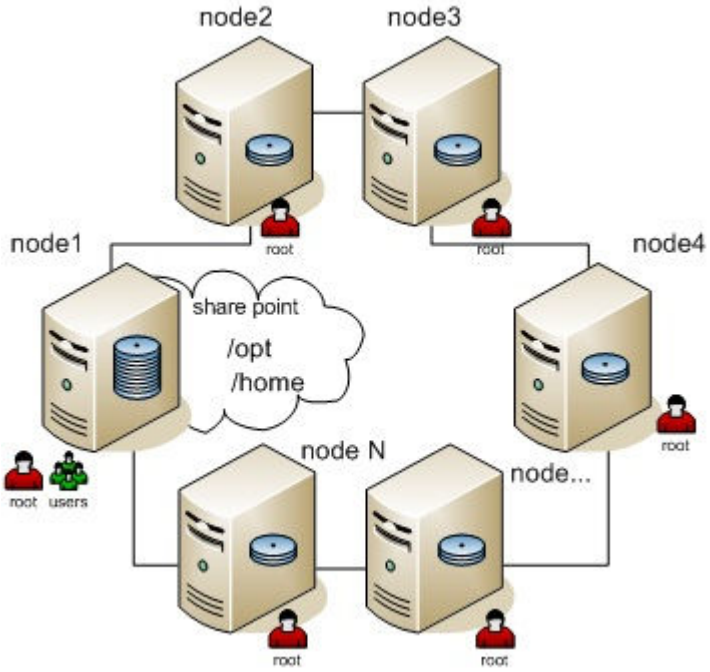
   The `insmod-sep` script loads the driver into the system on the current node. The `boot-script` script configures the driver boot script and installs it in the appropriate system directory. Use the `--help` option to view details on the available script options.

After installation is complete, you can verify that the kernel driver is installed and loaded on a node using the following command:

```
lsmod | grep sep
```

Example

A homogeneous cluster system is set up where the users only have direct access to one node, node1. The only disk space available for writing is the user's home directory. The users have all of their data and software on the file system mounted on node1. They start their tasks using job scripts, which involve MPI mechanisms for dispatching the tasks among the other nodes.

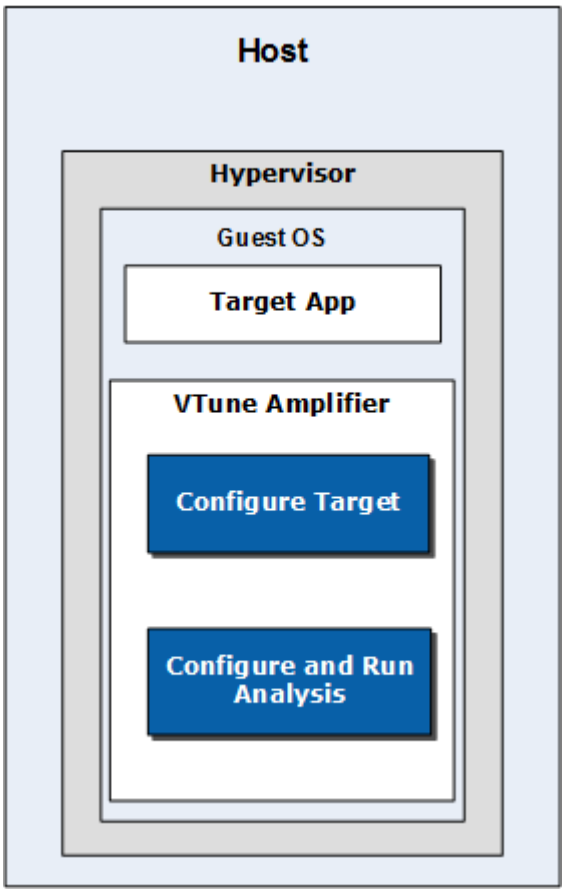


To allow users to run an analysis using VTune Amplifier, the administrator has to make sure that the product can be launched on each compute node. The kernel driver also must be installed and loaded on each compute node. After the product and drivers are installed, users can run performance collection on the nodes using the scheduling system scripts to launch an analysis. For example, for MPI applications the `mpiexec` script can be used on node1 to launch the profiling collector on the other nodes by specifying a user application to run as a parameter.

Installing Intel® VTune™ Amplifier for Use with a Virtual Machine

It is possible to collect data on a virtual machine for analysis with Intel VTune Amplifier. In most cases, VTune Amplifier is installed on the guest system using the installer user interface or command line installer. The data collection and analysis with VTune Amplifier is run on the virtual machine. For KVM and XEN Project*, the analysis is run using Perf*-based event-based sampling collection. For all other supported VMMs, it is run using the installed VTune Amplifier drivers. For more information about supported virtual machines and cloud service providers, see Targets in Virtualized Environments.

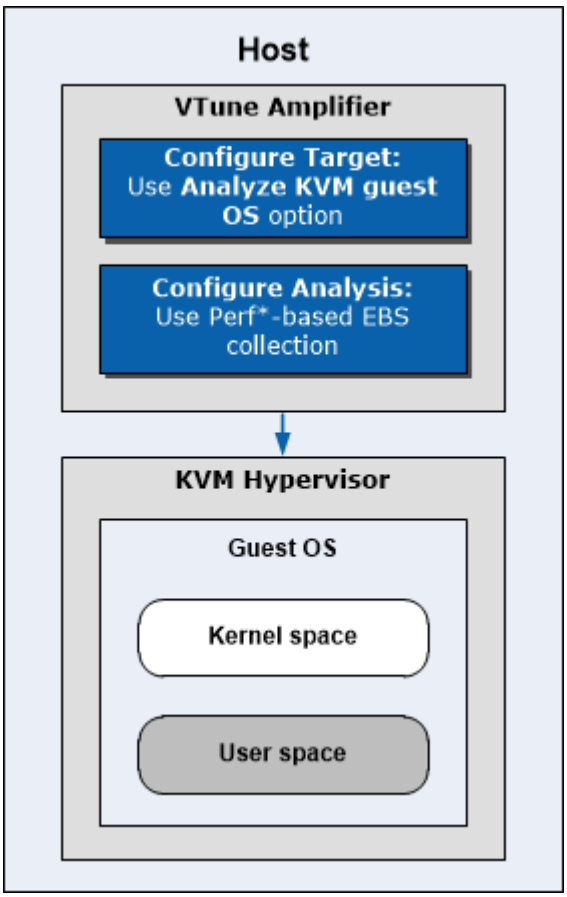
Tip
Intel VTune Amplifier may not run if it is not installed on a non-privileged guest OS. Refer to the documentation for your VMM to learn how to set up a privileged system.



Installing VTune Amplifier on the Host System to Analyze a KVM Guest System

You can collect analysis results for a guest system from VTune Amplifier installed on a host system if you are using KVM with a Linux* host and Linux guest. In this scenario, you would install VTune Amplifier on the host system using the installer user interface or command line installer.

NOTE
Symbols from the user space cannot be resolved by VTune Amplifier. Therefore, you cannot profile an application that is in the guest system from the host system.



Enabling Analysis on the Virtual Machine

Additional steps are required to enable performance analysis on the virtual machine.

- Enable VMWare* Analysis
- Enable KVM Analysis
- Enable XEN Project* Analysis

Not all analysis types and data can be collected on a virtual machine. Availability of data depends on which collectors are virtualized by the virtual machine manager. Additional information is available from Targets in Virtualized Environments and from the documentation for your virtual machine manager.

4 Post-Installation Steps

The following sections detail the steps required to configure your Intel® VTune™ Amplifier installation.

- Preparing a Target Linux* System for Remote Analysis
- Preparing a Target Embedded Linux* System for Remote Analysis
- Preparing a Target FreeBSD System for Remote Analysis
- Preparing a Target Android System for Remote Analysis
- Using Intel VTune Amplifier with a Virtual Machine
- Using Intel VTune Amplifier in a Cloud Environment
- Configuring SSH Access for Remote Collection
- Configuring a System for GPU Application Analysis
- Verifying VTune Amplifier Installation on a Linux* System
- Installing Command Line Collectors without a License
- Sampling Drivers
- Platform Profiler Setup (Preview)

The table below details the suggested reading paths based on your analysis needs:

Analyze performance on a remote Linux* system	Install the remote collectors and configure SSH access on a remote Linux system. If the remote collector installation fails to install the appropriate sampling drivers, the drivers can be installed manually. 1. Configuring SSH Access for Remote Collection 2. Preparing a Target Linux System for Remote Analysis 3. Build the Sampling Driver (optional)
Analyze performance on a remote Android* system	Configure the Android device for analysis and connect via ADB. If the appropriate sampling drivers are not available, the drivers can be installed manually. Compile the Android application for analysis. Specify the project search directories. 1. Preparing a Target Android System for Remote Analysis 2. Building and Installing the Sampling Driver for Android Targets (optional) 3. Preparing an Android Application for Analysis 4. Search Directories for Android
Analyze performance on a FreeBSD* system	Install VTune Amplifier collectors and drivers on a FreeBSD system. A FreeBSD license for Intel® System Studio is required. For more information, see Preparing a Target FreeBSD System for Remote Analysis.
Analyze performance on embedded Linux systems	Review the list of supported embedded Linux systems. Configure the embedded Linux environment for performance analysis. Begin with Preparing a Target Embedded Linux* System for Remote Analysis.

Monitor a virtual machine	Review the list of supported virtual machine managers and analysis type limitations. Configure the virtual machine for performance analysis. Begin with Using Intel VTune Amplifier with a Virtual Machine.
Analyze a target installed on the cloud	Review the list of supported cloud environments and learn how to configure the cloud target for remote analysis. For more information, see Using Intel VTune Amplifier in a Cloud Environment

Set Up Linux* System for Remote Analysis

You can collect data remotely on a target Linux* system by specifying the system as the analysis target in Intel® VTune™ Amplifier by selecting **Remote Linux (SSH)** in the **Where** pane when configuring an analysis. VTune Amplifier automatically installs the appropriate collectors on the target system. Specify a location for the install using the **VTune Amplifier installation directory on the remote system** field.

> **NOTE**
> The automatic installation on the remote Linux system does not build the sampling drivers although you can install the pre-built sampling drivers if you connect via password-less SSH as the root user. Driverless sampling data collection is based on the Linux Perf* tool functionality, which is available without Root access and has a limited scope of analysis options. To collect advanced hardware event-based sampling data, manually install the sampling driver or set up the password-less SSH connection with the Root user account.

If the collectors are not automatically installed or you get an error message after an automatic install attempt, use the following steps to manually prepare for data collection on a remote Linux system:

1. Install the VTune Amplifier collector on the target system.
2. Build and install sampling drivers. (Optional).
3. Set up an SSH access to the target system.
4. Set up the analysis target in VTune Amplifier.

Install the VTune Amplifier Collectors on the Target Device Manually

Use the following steps to set up analysis on a target regular or embedded Linux target system.

1. Copy the required target package archive to the target device using ftp, sftp, or scp. The following target packages are available on the host system where the VTune Amplifier is installed:
   - `<install_dir>/target/linux/vtune_amplifier_target_sep_x86.tgz` - provides hardware event-based sampling collector only (SEP) for x86 systems
   - `<install_dir>/target/linux/vtune_amplifier_target_sep_x86_64.tgz` - provides hardware event-based sampling collector only (SEP) for 64-bit systems
   - `<install_dir>/target/linux/vtune_amplifier_target_x86.tgz` - provides all VTune Amplifier collectors for x86 systems
   - `<install_dir>/target/linux/vtune_amplifier_target_x86_64.tgz` - provides all VTune Amplifier collectors for 64-bit systems

   > **NOTE**
   > Use both `*_x86` and `*_x86_64` packages if you plan to run and analyze 32-bit processes on 64-bit systems.

2. On the target device, unpack the product package to the `/tmp` directory or another writable location on the system:

```
target> tar -zxvf <target_package>.tgz
```

VTune Amplifier target package is located in the newly created directory `/tmp/vtune_amplifier_<version>.<package_num>`.

When collecting data remotely, the VTune Amplifier looks for the collectors on the target device in its default location: `/tmp/vtune_amplifier_<version>.<package_num>`. It also temporary stores performance results on the target system in the `/tmp` directory. If you installed the target package to a different location or need to specify another temporary directory, make sure to configure your target properties in the **Configure Analysis** window as follows:

- Use the **VTune Amplifier installation directory on the remote system** option to specify the path to the VTune Amplifier on the remote system. If default location is used, the path is provided automatically.
- Use the **Temporary directory on the remote system** option to specify a non-default temporary directory.

Alternatively, use the`-target-install-dir` and `-target-tmp-dir` options from the `amplxe-cl` command line.

Build and Install the Drivers Manually

NOTE
Building the sampling drivers is only required if the drivers were not built as part of the collector installation. The installation output should inform you if building the sampling driver is required.

To enable hardware event-based sampling analysis on your target device:

1. Build the sampling driver on the target system.

   NOTE

   - Make sure kernel headers correspond to the kernel version running on the device. For details, see the `README.txt` file in the `sepdk/src` directory.
   - Make sure compiler version corresponds to the architecture (x86 or x86_64) of the kernel running on the target system.
   - For Hotspots in hardware event-based sampling mode, Microarchitecture Exploration, and Custom event-based sampling analysis types, you may not need root credentials and installing the sampling driver for systems with kernel 2.6.32 or higher, which exports CPU PMU programming details over `/sys/bus/event_source/devices/cpu/format` file system. Your operating system limits on the maximum amount of files opened by a process as well as maximum memory mapped to a process address space still apply and may affect profiling capabilities. These capabilities are based on Linux Perf* functionality and all its limitations fully apply to the VTune Amplifier as well. For more information, see the *Tutorial: Troubleshooting and Tips* topic at https://perf.wiki.kernel.org/index.php/Main_Page.

2. On the target device, install the drivers.

   If the insmod-sep script does not work on the target system due to absence of standard Linux commands, you may install drivers manually using the Linux OS `insmod` command directly.

   NOTE
   To build the sampling driver as RPM using build services as Open Build Service (OBS), use the `sepdk.spec` file located at the `<install_dir>/sepdk/src` the directory.

Set up SSH Access

After installing the collectors and ensuring that the appropriate drivers are installed, set up SSH access to the target system.

Set up Analysis Target

After completing all other configuration steps for the remote Linux system, you can run an analysis using VTune Amplifier. Before running the first analysis, you must set up the analysis target.

Enable Linux* Kernel Analysis

For successful performance analysis of the kernel and system libraries, do the following:

1. Enable kernel modules resolution.
2. Download and install debug info packages available for your Linux system version.
3. Build the Linux kernel with debug information.

Enable Kernel Modules Resolution

To provide accurate performance statistics for the Linux kernel, the VTune Amplifier requires kernel modules information provided in the `/proc/kallsyms` file. Make sure the `/proc/sys/kernel/kptr_restrict` file contains values that enable reading `/proc/kallsyms` and providing non-zero addresses for the kernel pointers:

- If the `kptr_restrict` value is 0, kernel addresses are provided without limitations (recommended).
- If the `kptr_restict` value is 1, addresses are provided if the current user has a `CAP_SYSLOG` capability.
- If the `kptr_restrict` value is 2, the kernel addresses are hidden regardless of privileges the current user has.

See more details at: http://lwn.net/Articles/420403/, http://man7.org/linux/man-pages/man7/capabilities.7.html.

If kernel pointers information was explicitly hidden by setting the `kptr_restrict` to a non-zero value, hardware event-based analysis results may not contain functions from kernel modules. As a result, you may see the CPU time associated with the `[Outside any known module]` item. To workaround this problem for the current session, set the contents of the `/proc/sys/kernel/kptr_restrict` file to 0 before starting the VTune Amplifier as follows:

```
sysctl -w kernel.kptr_restrict=0
```

To resolve symbols for the Linux kernel, the VTune Amplifier also uses the `System.map` file created during the kernel build and shipped with the system by default. If the file is located in a non-default directory, you may add it to the list of search directories in the **Binary/Symbol Search** dialog box when configuring your target properties.

> **NOTE**
> The settings in the `/proc/kallsyms` and `System.map` file enable the VTune Amplifier to resolve kernel symbols and view kernel functions and kernel stacks but do not enable the assembly analysis.

Download and Install Available Debug Kernel Versions

After installing the Linux operating system, the kernel is contained in `vmlinux`, or `vmlinuz`, or `bzImage` in `/boot`. Linux vendors typically release compressed kernel files stripped of symbols (`vmlinuz` or `bzImage`). `vmlinux` is the uncompressed Linux kernel, but it does not include debug information. So, by default the

VTune Amplifier cannot retrieve kernel function information from these kernels and presents all hot addresses captured in the kernel as a unique function or module named `[vmlinux]`. However, some vendors have released special debug versions of their kernels that are suitable for performance analysis.

1. Use the `uname -r` command to identify the running Linux kernel version.
2. Download and install two RPMs matching your system: `kernel-debug-debuginfo-*.rpm` and `kernel-debuginfo-common-*.rpm`. To do this, use any of the following options:
   - Browse through the RPMs on your installation CDs or DVDs. For example, for SuSE Linux Enterprise* 9, 10, and 11 distros, SuSE provides "debug" kernel RPMs (`kernel-debug-*.rpm`) available on the install CD or from the website. After installing the RPM, the debug version of the kernel file is located under `/boot/vmlinux-*-debug` or under `/boot/vmlinuz-*-debug`. You need to manually uncompress this kernel file using the `gunzip` program.
   - Browse through the OS vendor FTP site and download the packages. For example: look at ftp://ftp.redhat.com/pub/redhat/linux/enterprise/5Server/en/os to get packages for Redhat* Enterprise Server.
   - Look for other sources on the internet. For example, for Red Hat Enterprise* Linux 3, 4 and 5 distros, Red Hat provides `debuginfo` RPMs at http://people.redhat.com/duffy/debuginfo/. After installing the RPM, the debug version of the kernel file is located under `/usr/lib/debug/boot` (EL 3) or `/usr/lib/debug/lib/modules` (EL 4, 5).
3. Use the following commands to install the RPMs:

```
rpm -ivh kernel-debuginfo-common-*.rpm
rpm -ivh kernel-debug-debuginfo-*.rpm
```

   For some operating systems, you can use `yum` to install packages directly, for example:

```
yum --enablerepo=rhel-debuginfo install kernel-debuginfo
```

4. Verify that the packages have been installed, for example:

```
rpm -qa|grep kernel
```

5. Modify the VTune Amplifier target properties and specify the path to the uncompressed kernel binary in the Binary/Symbol Search dialog box, for example: `/usr/lib/debug/lib/modules/2.6.18-128.el5debug/`.

Build the Linux Kernel with Debug Information

1. Configure the kernel sources.
2. Edit the kernel source top-level Makefile and add the `-g` option to the following variables:

```
CFLAGS_KERNEL := -g

CFLAGS := -g
```

3. Run `make clean; make` to create the `vmlinux` kernel file with debug information. Once a debug version of the kernel is created or obtained, specify that kernel file as the one to use during performance analysis.

As soon as the debug information is available for your kernel modules, any future analysis runs will display the kernel functions appropriately. To resolve the previously collected data against this new symbol information, update the project Search Directories and click the **Re-resolve** button to apply the changes.

Embedded Linux* Targets

Use the Intel® VTune™ Amplifier for performance analysis on Embedded Linux systems, Wind River*, Yocto Project*, FreeBSD* and others.*

Embedded device performance data can be collected remotely on the embedded device and running the analysis from an instance of VTune Amplifier installed on the host system. This is useful when the target system is not capable of local data analysis (low performance, limited disk space, or lack of user interface control).

NOTE
Root access to the operating system kernel is required to install the collectors and drivers required for performance analysis using VTune Amplifier.

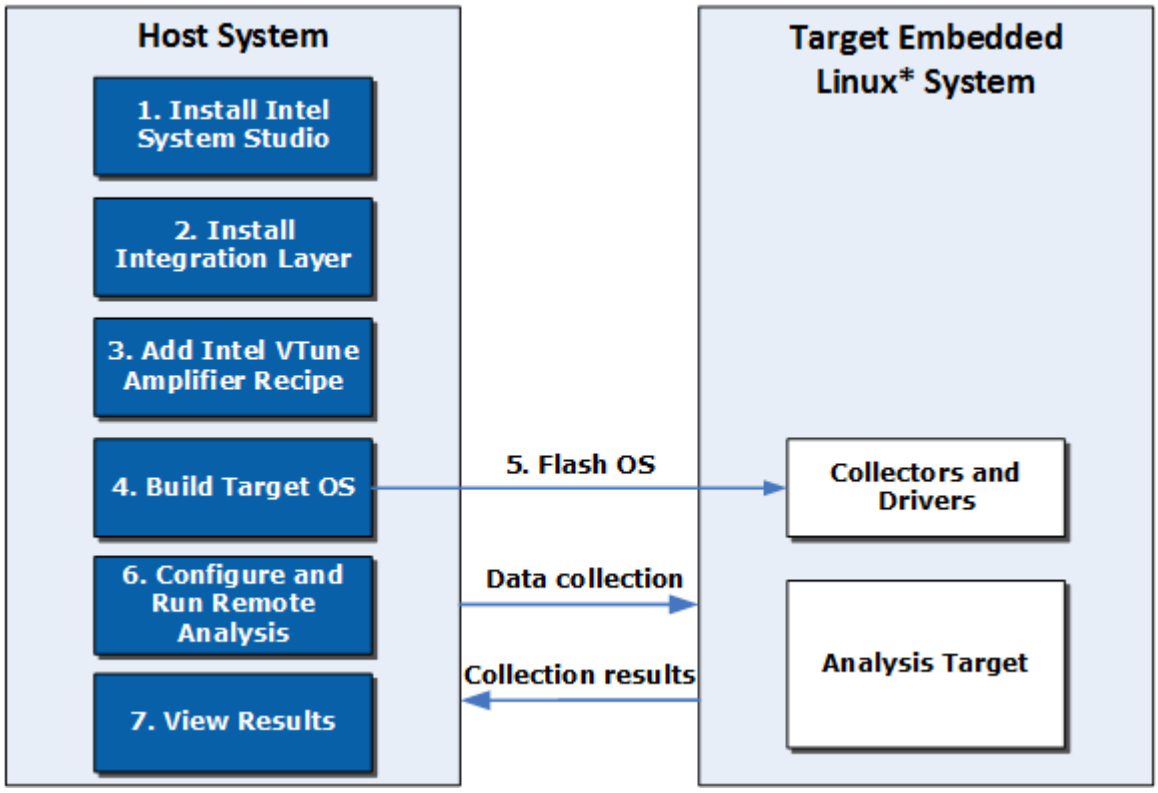
There are multiple ways to enable performance analysis on an embedded device:

- Using the Intel System Studio integration layer (Wind River* Linux and Yocto Project* only)
- Using the Intel VTune Amplifier Yocto Project Integration Layer
- Using the bundled VTune Amplifier installation packages

Using the Intel System Studio Integration Layer

NOTE
The Intel System Studio integration layer works for embedded systems with Wind River Linux or Yocto Project installed.

The Intel System Studio integration layer allows the Intel System Studio products to be fully integrated with a target operating system by building the drivers and corresponding target packages into the operating system image automatically. Use this option in the case where a platform build engineer has control over the kernel sources and signature files, but the application engineer does not. The platform build engineer can integrate the product drivers with the target package and include them in the embedded device image that is delivered to the application engineer.



1. Install Intel System Studio.	Install Intel System Studio using the installer GUI.
2. Install the Intel System Studio integration layer.	1. Copy the integration layer from the Intel System Studio installation folder to the target operating system development folder. 2. Run the post-installation script: `<iss-install-dir>/YoctoProject/meta-intel-iss/yp-setup/postinst_<OS>_iss.sh <ISS_BASE_dir>` For example, for Wind River Linux: `/YoctoProject/meta-intel-iss/yp-setup/postinst_wr_iss.sh`
3. Build the recipe that includes the appropriate VTune Amplifier package.	1. Add the path to the `/YoctoProject/meta-intel-iss` to the `bblayers.conf` file: `BBLAYERS= "\` `...` `<OS_INSTALL_DIR>/YoctoProject/meta-intel-iss\` `...` `"` 2. Add the VTune Amplifier recipes to `conf/local.conf`. Possible recipes include: • `intel-vtune-drivers`: integrates all VTune Amplifier drivers for PMU-based analysis with stacks and context switches. Requires additional kernel options to be enabled. • `intel-vtune-sep-driver`: integrates drivers for PMU-based analysis with minimal requirements for kernel options. For more information about these collection methods, see *Remote Linux Target Setup* in the VTune Amplifier help.
4. Build the target operating system.	Build the target operating system, which will complete the integration of the VTune Amplifier collectors and drivers.
5. Flash the operating system to the target embedded device.	After flashing the operating system to the target embedded device, ensure that the appropriate VTune Amplifier drivers are present. For more information, see Building the Sampling Drivers for Linux Targets.
6. Run the analysis on the target embedded device.	Run the analysis is from the host system using an SSH connection or using the SEP commands. Use the following steps to run the analysis from the host system: 1. Set up a password-less SSH access to the target using RSA keys. 2. Specify your target application and remote system. **NOTE** After configuring the remote connection, VTune Amplifier will install the appropriate collectors on the target system. 3. Choose an analysis type. 4. Run the analysis from the host.

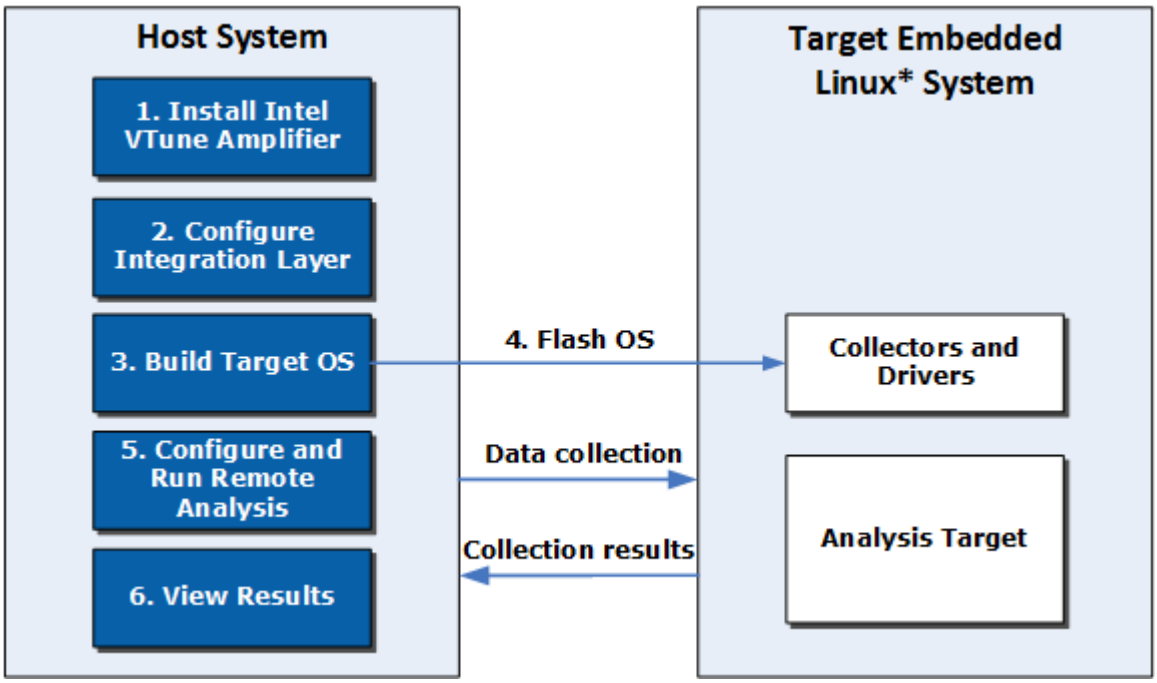
	Use the information available in the *Sampling Enabling Product User's Guide* to run the SEP commands.
7. View results in the VTune Amplifier GUI	View the collected data on the host.

Examples

Configuring Yocto Project* with the Intel System Studio Integration Layer

Using the Intel VTune Amplifier Yocto Project Integration Layer

The Intel VTune Amplifer Yocto Project integration layer builds the drivers into the operating system image automatically. Use this option in the case where a platform build engineer has control over the kernel sources and signature files, but the application engineer does not. The platform build engineer can integrate the product drivers with the target package and include them in the embedded device image that is delivered to the application engineer.



1. Install Intel VTune Amplifier.	Install Intel VTune Amplifier.
2. Configure the integration layer.	**1.** Extract the `<install-dir>/target/linux/vtune_amplifier_target_x86.tgz` or `<install-dir>/target/linux/vtune_amplifier_target_x86_64.tgz` package. **2.** Modify the `sepdk/vtune-layer/conf/user.conf` file to specify user settings. **a.** Specify one of the following paths: • Path to unzipped target package: `VTUNE_TARGET_PACKAGE_DIR = "<PATH>"` • Path to VTune Amplifier installation directory: `VTUNE_AMPLIFIER_DIR = "<PATH>"`

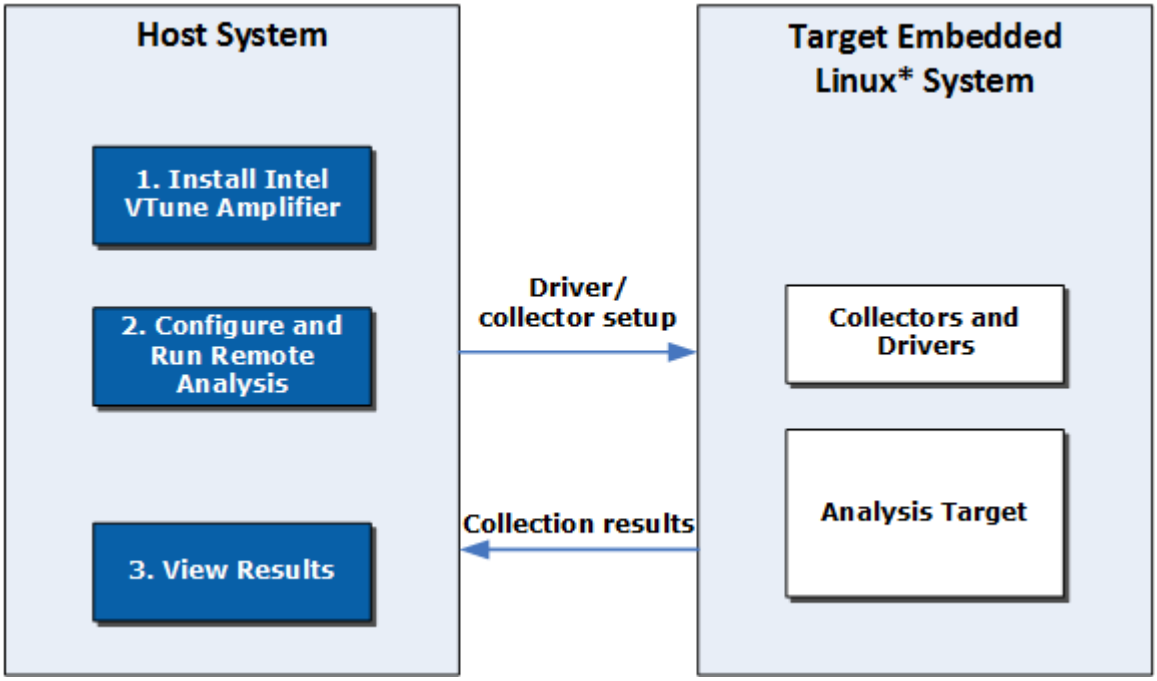
	b. (Optional) To integrate the SEP driver during system book, specify `ADD_TO_INITD = "y"`. **3.** Copy the integration layer to the Yocto Project development environment. **4.** Add the path to the layer to the `bblayers.conf` file: `BBLAYERS= "\` `...` `<OS_INSTALL_DIR>/vtune-layer\` `...` `"` **5.** Add the VTune Amplifier recipes to `conf/local.conf`. Possible recipes include: • `intel-vtune-drivers`: integrates all VTune Amplifier drivers for PMU-based analysis with stacks and context switches. Requires additional kernel options to be enabled. • `intel-vtune-sep-driver`: integrates drivers for PMU-based analysis with minimal requirements for kernel options. For more information about these collection methods, see *Remote Linux Target Setup* in the VTune Amplifier help.
3. Build the target operating system.	Build the target operating system, which will complete the integration of the VTune Amplifier collectors and drivers.
4. Flash the operating system to the target embedded device.	After flashing the operating system to the target embedded device, ensure that the appropriate VTune Amplifier drivers are present. For more information, see Building the Sampling Drivers for Linux Targets.
5. Run the analysis on the target embedded device.	Run the analysis is from the host system using an SSH connection or using the SEP commands. Use the following steps to run the analysis from the host system: **1.** Set up a password-less SSH access to the target using RSA keys. **2.** Specify your target application and remote system. **3.** Choose an analysis type. **4.** Run the analysis from the host. Use the information available in the *Sampling Enabling Product User's Guide* to run the SEP commands.
6. View results in the VTune Amplifier GUI	View the collected data on the host.

Example

Configuring Yocto Project with the VTune Amplifier Integration Layer

Using the Bundled Intel VTune Amplifier Installation Packages

You can build the appropriate drivers and install the VTune Amplifier collectors on your kernel image manually with a command line. This option requires root access to the configured kernel source.



1. Install Intel VTune Amplifier.	Install using the installer GUI.
2. Run the analysis on the target embedded device.	Use the following steps to run the analysis from the host system: **1.** Set up a password-less SSH access for the root user to the target using RSA keys. **2.** Specify your target application and remote system. The collectors and drivers within the package should be automatically installed. **3.** Choose an analysis type. **4.** Run the analysis from the host.
3. View results in the VTune Amplifier GUI	View the collected data on the host.

Troubleshooting

If the drivers were not built during collector installation, the installation output should inform you that building the sampling driver is required.

The drivers are built either on the target system or on the host system, depending on compiler toolchain availability:

1. If the compiler toolchain is available on the target system:
   a. On the target embedded device, build the driver from the `<install-dir>/sepdk/src` directory using the `./build-driver` command.
   b. Load the driver into the kernel using the `./insmod-sep` command.
2. If the compiler toolchain is not available on the target system:
   a. On the host system, cross-build the driver using the driver source from the target package `sepdk/src` directory with the `./build-driver` command. Provide the cross-compiler (if necessary) and the target kernel source tree for the build.
   b. Copy the `sepdk/src` folder to the target system.
   c. Load the driver into the kernel using the `./insmod-sep` command.

For more information, see Building the Sampling Drivers for Linux Targets.

Example

Configuring Yocto Project with Intel VTune Amplifier Target Packages

Configure Yocto Project* and VTune™ Amplifier with the VTune Amplifier Integration Layer

Intel® VTune™ Amplifier can collect and analyze performance data on embedded Linux* devices running Yocto Project*. This topic provides an example of setting up the VTune Amplifier to collect performance data on an embedded device with Yocto Project 1.8 installed using the Intel VTune Amplifier integration layer provided with the product installation files. The process integrates the VTune Amplifier product drivers with the target package and includes them in the embedded device image. Root access to the kernel is required.

> **NOTE**
> VTune Amplifier is able to collect some performance data without installing the VTune Amplifier drivers. To collect driverless event-based sampling data, installing the drivers and root access is not required. For full capabilities, install the VTune Amplifier drivers as described here.

Select the Target Package

VTune Amplifier provides two Yocto Project recipes in the following packages:

- The `vtune_amplifier_target_sep_x86_64.tgz` package includes the `intel-vtune-sep-driver` recipe, which enables performance data collection using hardware event-based sampling. Attempting to collect stacks when using this recipe will automatically switch to driverless collection mode. This recipe has minimal requirements for Linux kernel configuration.
- The `vtune_amplifier_target_x86_64.tgz` package includes the `intel-vtune-drivers` recipe, which enables the full performance data capabilities using hardware event-based sampling. This recipe has additional requirements for Linux kernel configuration. The `intel-vtune-drivers` recipe is a superset of the `intel-vtune-sep-driver` recipe.

Only one recipe can be used at a time. There is no difference between the `x86` and `x86_64` target packages for building recipes within Yocto Project. Both can be used on either 32 bit or 64 bit systems.

1. Download the VTune Amplifier target package or locate the package in the `<install-dir>/target/linux` directory on the host system where VTune Amplifier is installed.
2. Copy the selected target package to a location on the Yocto Project build system.

Prepare the Integration Layer

1. On the Yocto Project build system, extract the `vtune_amplifier_target_sep_x86_64.tgz` or `vtune_amplifier_target_x86_64.tgz` archive to a writeable location.

```
cd $HOME
tar xvzf vtune_amplifier_target_x86_64.tgz
```

2. (Optional) Modify the `$HOME/vtune_amplifier_<version>/sepdk/vtune-layer/conf/user.conf` file to specify user settings.
   a. If the VTune Amplifier recipe has been split from the target package, specify one of the following paths:
      - Path to unzipped target package: `VTUNE_TARGET_PACKAGE_DIR = "$HOME/vtune_amplifier_<version>"`
      - Path to VTune Amplifier: `VTUNE_AMPLIFIER_DIR = "/opt/intel/vtune_amplifier"`
   b. To integrate the SEP driver during system boot, specify `ADD_TO_INITD = "y"`.

3. In the Yocto Project development environment, add the path to the layer to the `bblayer.conf` file. For example:

```
vi conf/bblayers.conf
BBLAYERS = "$HOME/vtune_amplifier_<version>/sepdk/vtune-layer\"
```

   Your file should look similar to the following:

```
BBLAYERS ?= " \
  $HOME/source/poky/meta \
  $HOME/source/poky/meta-poky \
  $HOME/source/poky/meta-yocto-bsp \
  $HOME/source/poky/meta-intel \
  $HOME/vtune_amplifier_2019/sepdk/vtune-layer \
  "
```

4. Specify the Intel VTune Amplifier recipe in `conf/local.conf`. In this example, the `intel-vtune-drivers` is used.

```
vi "conf/local.conf"
IMAGE_INSTALL_append = " intel-vtune-drivers"
```

NOTE
You cannot add both `intel-vtune-drivers` and `intel-vtune-sep-driver` at the same time.

Build and Flash the Target Operating System

1. Build the target operating system. For example:

```
bitbake core-image-sato
```

NOTE
If you modified the kernel configuration options, make sure the kernel is recompiled.

2. Flash the operating system to the embedded device.

Configure and Run Remote Analysis

Use the following steps on the host system to set up and launch the analysis on the embedded device:

1. Set up a password-less SSH access to the target using RSA keys.
2. Create a new project.
3. Select the **remote Linux (SSH)** analysis target and specify the collection details.
4. Configure the analysis type.
5. Start the analysis.

Configure Yocto Project*/Wind River* Linux* and Intel® VTune™ Amplifier with the Intel System Studio Integration Layer

You can use Intel® VTune™ Amplifier to collect and analyze performance data on embedded Linux* devices running Yocto Project* or Wind River* Linux*. This example describes how you set up VTune Amplifier using the Intel System Studio integration layer, to collect performance data on an embedded device with Yocto Project 1.8 or Wind River* Linux* installed. The integration layer is available with the product installation files. The process integrates the VTune Amplifier product drivers with the target package and includes them in the embedded device image. For this example, you need root access to the kernel.

Install the Intel System Studio Integration Layer

Prerequisite: Install Intel System Studio on the host system.

1. Copy the integration layer from the Intel System Studio installation folder to the appropriate development folder.

   For Yocto Project*:

   ```
   cp -r <ISS_BASE_DIR>/YoctoProject/meta-intel-iss <YOCTO_HOME>/
   ```

   For Wind River* Linux*:

   ```
   cp -r <ISS_BASE_DIR>/YoctoProject/meta-intel-iss <WR_HOME>/
   ```

   where

   - `<ISS_BASE_DIR>` : Root folder of the Intel System Studio installation. By default, this is `/opt/intel/system_studio_<version>.x.y/`. For example, for the 2019 version, the root folder is `/opt/intel/system_studio_2019.0.0/`.
   - `<YOCTO_HOME>` : Root folder of the Yocto Project* cloned directory.
   - `<WR_HOME>` : Root folder of the Wind River* Linux* cloned directory.
2. Register the layer by running the post-installation script.

   For Yocto Project*:

   In the shell console, go to the `<YOCTO_HOME>` folder and run this command:.

   ```
   $ meta-intel-iss/yp-setup/postinst_yp_iss.sh <ISS_BASE_DIR>
   ```

   For Wind River* Linux*:

   In the shell console, go to the `<WR_HOME>` folder and run this command:.

   ```
   $ meta-intel-iss/yp-setup/postinst_wr_iss.sh <ISS_BASE_DIR>
   ```

To uninstall the Intel System Studio integration:

1. Run the appropriate script to uninstall:

   For Yocto Project*:

   In the shell console, go to the `<YOCTO_HOME>` folder and run this command:.

   ```
   $ meta-intel-iss/yp-setup/uninst_yp_iss.sh
   ```

   For Wind River* Linux*:

   In the shell console, go to the `<WR_HOME>` folder and run this command:.

   ```
   $ meta-intel-iss/yp-setup/uninst_wr_iss.sh
   ```
2. Remove the `meta-intel-iss` layer.

Add the Intel VTune Amplifier Recipe

1. Add the path to the `wr-iss-<version>` to the `bblayer.conf` file. For example:

   ```
   vi /path/to/poky-fido-10.0.0/build/conf/bblayers.conf
   BBLAYERS = "$HOME/source/poky/wr-iss-2019\"
   ```

   Your file should look similar to the following:

   ```
   BBLAYERS ?= " \
     $HOME/source/poky/meta \
     $HOME/source/poky/meta-poky \
     $HOME/source/poky/meta-yocto-bsp \
   ```

```
  $HOME/source/poky/meta-intel \
  $HOME/source/poky/wr-iss-2019 \
  "
```

2. Add the Intel VTune Amplifier recipe to `conf/local.conf`. Two recipes are available, `intel-vtune-drivers` and `intel-vtune-sep-driver`. In this example, the `intel-vtune-drivers` is used so the analysis can be run from the VTune Amplifier GUI on the host system.

```
vi "conf/local.conf"
IMAGE_INSTALL_append = " intel-vtune-drivers"
```

NOTE
You cannot add both `intel-vtune-drivers` and `intel-vtune-sep-driver` at the same time.

Build and Flash the Target Operating System

1. Build the target operating system. For example:

```
bitbake core-image-sato
```

2. Flash the operating system to the embedded device.

Configure and Run Remote Analysis

Use the following steps on the host system to set up and launch the analysis on the embedded device:

1. Set up a password-less SSH access to the target using RSA keys.
2. Create a new project.
3. Select the **remote Linux (SSH)** analysis target and specify the collection details.
4. Configure the analysis type.
5. Start the analysis.

Configure Yocto Project* and Intel® VTune™ Amplifier with the Linux* Target Package

Intel® VTune™ Amplifier can collect and analyze performance data on embedded Linux* devices. This topic provides an example of setting up Intel VTune Amplifier to collect performance data on an embedded device running Yocto Project*. The first section provides information for a typical use case where the required collectors are automatically installed. The second section provides steps to manually install the collectors and the VTune Amplifier drivers for hardware event-based sampling data collection.

Automatically Configure and Run Remote Analysis

Use the following steps on the host system to set up and launch the analysis on the embedded device:

1. Set up a password-less SSH access to the target using RSA keys.
2. Open VTune Amplifier and create a new project.
3. Select the **remote Linux (SSH)** analysis target and specify the collection details. VTune Amplifier connects to the target system and installs the appropriate collectors. If the automatic installation fails or if you want to collect hardware event-based sampling with the VTune Amplifier drivers, follow the instructions below to manually configure the target system.
4. Select the analysis type.
5. Start the analysis.

Manually Configure the Linux Target System

Use these steps only if the automatic installation fails.

1. Copy the target package archive to the target device. The following target packages are available:

- `<install-dir>/target/vtune_amplifier_target_sep_x86.tgz` - provides hardware event-based sampling collector only (SEP) for x86 systems
- `<install-dir>/target/vtune_amplifier_target_x86.tgz` - provides all VTune Amplifier collectors for x86 systems
- `<install-dir>/target/vtune_amplifier_target_sep_x86_64.tgz` - provides hardware event-based sampling collector only (SEP) for 64-bit systems
- `<install-dir>/target/vtune_amplifier_target_x86_64.tgz` - provides all VTune Amplifier collectors for 64-bit systems

For example, the following command copies the `vtune_amplifier_target_x86_64.tgz` package to the embedded device using SCP:

```
> scp -r vtune_amplifier_target_x86_64.tgz root@123.45.67.89:/opt/intel/
```

2. Extract the file on the target system. For example:

```
> tar -xvsf vtune_amplifier_target_x86_64.tgz
```

3. Make sure the sampling driver is available on the target system. The installation output should inform you if building the sampling driver is required. If it is not, you will need to build the sampling driver and install it on the target system.

   If the compiler toolchain is available on the target embedded system, build the driver on the target device using the following steps:

   a. Open a command prompt and navigate to the `<install-dir>/sepdk/src` directory. For example:

```
> cd /opt/intel/vtune_amplifier_2019.0.0.0/sepdk/src
```

   b. Build the driver using the `./build-driver` command. For example:

```
> ./build-driver -ni \ --kernel-src-dir=/usr/src/kernel/ \ --kernel-version=4.4.3-yocto-standard
\ --make-args="PLATFORM=x64 ARITY=smp"
```

   c. Load the driver into the kernel using the `./insmod-sep` command.

   If the compiler toolchain is not available on the target embedded system, build the driver on the host system and install it on the target device using the following steps:

   a. Open a command prompt and navigate to the `<install-dir>/sepdk/src` directory. For example:

```
> cd /opt/intel/vtune_amplifier_2019.0.0.0/sepdk/src
```

   b. Cross-build the driver using the using the `./build-driver` command. Provide the cross-compiler (if necessary) and the target kernel source tree for the build. For example:

```
> mkdir drivers
>./build-driver -ni \
--c-compiler=i586-i586-xxx-linux-gcc \
--kernel-version=4.4.3-yocto-standard \
--kernel-src-dir=/usr/src/kernel/ \
--make-args="PLATFORM=x32 ARITY=smp" \
--install-dir=./drivers
```

   c. Copy the `sepdk/src/drivers` folder to the target system.
   d. Load the driver into the kernel using the `./insmod-sep` command.

Set Up FreeBSD* System

Intel® VTune™ Amplifier allows you to collect performance data remotely on a FreeBSD target system.*

Intel® VTune™ Amplifier includes a target package for collecting event-based sampling data on a FreeBSD* target system either via the remote collection capability or by collecting the results locally on the FreeBSD system and copying them to a Linux*, Windows*, or macOS* host system. The collected data is then displayed on a host system that supports the graphical interface.

1. Install VTune Amplifier on your Linux, Windows, or macOS host.
2. Install the appropriate sampling drivers on the FreeBSD target system. Use the `<vtune-install-dir>/target/freebsd/vtune_amplifier_target_x86_64.tgz` file for analysis using VTune Amplifier or the `<vtune-install-dir>/target/freebsd/vtune_amplifier_target_sep_x86_64.tgz` file for analysis using the sampling enabling product (SEP) collectors.
3. Collect performance data using one of the following methods. For more information about each of these methods, see FreeBSD* Targets and Remote Linux Target Setup in the Intel VTune Amplifier online help.
   - Remote analysis from the host system using the VTune Amplifier command line or GUI.
   - Native analysis on the target system using the VTune Amplifier command line.
   - Native analysis on the target system using the SEP collectors.
4. Review the results on the host system.

Install the Sampling Drivers on FreeBSD

Use the following steps to configure your FreeBSD target system for event-based sampling analysis. Root privileges are required on the target system to install the VTune Amplifier drivers.

1. Copy the `<vtune-install-dir>/target/freebsd/vtune_amplifier_target_x86_64.tgz` file to the target system using ftp, sftp, or scp.
2. Extract the archive to the `/opt/intel` directory on the target system.
3. Navigate to the following location: `/opt/intel/sepdk/modules`
4. Run the following commands to build the appropriate drivers:

```
$ make
$ make install
```

5. Run the following command to install the drivers:

```
$ kldload sep pax
```

Allow non-root users to run an event-based sampling analysis by running the following commands after installing the drivers:

```
$ chgrp -R <user_group> /dev/pax
$ chgrp -R <user_group> /dev/sep
```

Remove the Sampling Drivers from FreeBSD

Run the following command to unload the sampling drivers:

```
$ kldunload sep pax
```

Set Up Android* System

When using the VTune Amplifier to collect data remotely on a target Android device, make sure to:

- Configure your Android device for analysis.
- Gain adb access to an Android device.
- For hardware event-based sampling, gain a root mode adb access to the Android device.
- Use the pre-installed drivers on the target Android system.

Optionally, do the following:

- Enable Java* analysis.
- To view functions within Android-supplied system libraries, device drivers, or the kernel, get access from the host development system to the exact version of these binaries with symbols not stripped.

- To view sources within Android-supplied system libraries, device drivers, or the kernel, get access from the host development system to the sources for these components.

Configure an Android Device for Analysis

To configure your Android device, do the following:

1. Allow Debug connections to enable adb access:
   a. Select **Settings** > **About <*device*>**.
   b. Tap **Build number** seven times to enable the **Developer Options** tab.
   c. Select the **Settings** > **Developer Options** and enable the **USB debugging** option.

   NOTE
   Path to the **Developer Options** may vary depending on the manufacture of your device and system version.

2. Enable **Unknown Sources** to install the VTune Amplifier Android package without Google* Play. To do this, select **Settings** > **Security** and enable the **Unknown Sources** option.

Gain ADB Access to an Android Device

VTune Amplifier collector for Android requires connectivity to the Android device via adb. Typically Android devices are connected to the host via USB. If it is difficult or impossible to get adb access to a device over USB, you may get adb over Ethernet or WiFi. To connect ADB over Ethernet or WiFi, first connect to Ethernet or connect to a WiFi access point and then do the following:

1. Find the IP Address of the target. The IP address is available in Android for Ethernet via **Settings>Wireless&Networks>Ethernet>IP Address** or for Wi-Fi via **Settings>Wireless&Networks>Wi-Fi><*Connected Access Point*>>IP Address**.
2. Make sure adb is enabled on the target device. If not enabled, go to Terminal App (of your choice) on the device and type:

```
> su
> setprop service.adb.tcp.port 5555
> stop adbd
> start adbd
```

3. Connect adb on the host to the remote device. In the Command Prompt or the Terminal on the host, type:

```
> adb connect <IPAddres>:5555
```

Gain a Root Mode ADB Access to the Android Device

For performance analysis on Android platforms, you typically need a root mode adb access to your device to:

- Install and load drivers needed for hardware event-based sampling.
- Enable the Android device to support Java* analysis.
- Run hardware event-based sampling analysis.

NOTE
There are several analysis types on Android systems that do NOT require root privileges such as Hotspots (user-mode samplingmode) and Perf*-based driverless sampling event-based collection.

Depending on the build, you gain root mode adb access differently:

- **User/Production builds** : Gaining root mode adb access to a user build of the Android OS is difficult and different for various devices. Contact your manufacturer for how to do this.

- **Engineering builds** : Root-mode adb access is the default for engineering builds. Engineering builds of the Android OS are by their nature not "optimized". Using the VTune Amplifier against an engineering build is likely to result in VTune Amplifier identifying code to optimize which is already optimized in user and userdebug builds.
- **Userdebug builds** : Userdebug builds of the Android OS offer a compromise between good results and easy-to-run tools. By default, userdebug builds run adb in user mode. VTune Amplifier tools require root mode access to the device, which you can gain via typing `adb root` on the host. These instructions are based on userdebug builds.

Use the Pre-installed Drivers on the Target Android System

For hardware event-based sampling analysis, the VTune Amplifier needs sampling drivers to be installed. On some versions of Android systems, including most of the Intel supplied reference builds for SDVs, the following drivers are pre-installed in `/lib/modules` or `/system/lib/modules` :

- Hardware event-based analysis collectors:
  - `socperf2_x.ko`
  - `pax.ko`
  - `sep3_x.ko`
  - `sep4_x.ko`
  - `vtsspp.ko`

Typically having pre-installed drivers is more convenient. You can check for pre-installed drivers by typing:

`adb shell ls [/lib/modules|/system/lib/modules]`

If the drivers are not available or the version does not match requirements, consider building and installing the drivers.

Enable Java* Analysis on Android* System

Explore configuration settings required to enable Java analysis with Intel® VTune™ Amplifier on an Android system:

- Enable Java analysis on rooted devices
- Enabling Java analysis for code generated with ART* compiler

Enabling Java Analysis on Rooted Devices

By default, the VTune Amplifier installs the remote collector on the target rooted Android devices with the `--jitvtuneinfo=src` option. To change the Java profiling option for rooted devices, you need to re-install the remote collector on the target manually using the `--jitvtuneinfo=[jit|src|dex|none]` option on (Windows) or `amplxe-androidreg.sh` (Linux). For example:

On Windows*:

`<install-dir>\bin32\amplxe-androidreg.bat --package-command=install --jitvtuneinfo=src`

On Linux*:

`<install-dir>/bin{32,64}/amplxe-androidreg.sh --package-command=install --jitvtuneinfo=src`

VTune Amplifier updates the `/data/local.prop` file as follows:

1. Basic information about the compiled trace: `root@android:/ # cat /data/local.prop dalvik.vm.extra-opts=-Xjitvtuneinfo:jit`
2. Mapping from JIT code to Java source code and basic information about the compiled trace: `root@android:/ # cat /data/local.prop dalvik.vm.extra-opts=-Xjitvtuneinfo:src`

3. Mapping from JIT code to DEX code and basic information about the compiled trace: `root@android:/ # cat /data/local.prop dalvik.vm.extra-opts=-Xjitvtuneinfo:dex`
4. JIT data collection. By default, JIT collection is disabled if you do not supply any options: `root@android:/ # cat /data/local.prop dalvik.vm.extra-opts=-Xjitvtuneinfo:none`

Additionally, if your Dalvik JVM supports instruction scheduling, disable it by adding `-Xnoscheduling` at the end of `dalvik.vm.extra-opts`. For example:

```
root@android:/ # cat /data/local.prop dalvik.vm.extra-opts=-Xjitvtuneinfo:src -Xnoscheduling
```

NOTE
Java analysis currently requires an instrumented Dalvik JVM. Android systems running on the 4th Generation Intel® Core™ processors or Android systems using ART vs. Dalvik for Java are not instrumented to support JIT profiling. You do not need to specify `--jitvtuneinfo=N`.

Tip
If you are able to see the `--generate-debug-info` option in the logcat output (`adb logcat *:S dex2oat:I`), the compiler uses this option.

Enabling Java Analysis for Code Generated with ART* Compiler

To enable a source-level analysis, the VTune Amplifier requires debug information for the analyzed binary files. By default, the ART compiler does not generate the debug information for Java code. Depending on your usage scenario, you may choose how to enable generating the debug information with the ART compiler:

NOTE
For releases prior to Android 6.0 Marshmallow*, the `--generate-debug-info` in the examples below should be replaced with `--include-debug-symbols`.

To Do This:	Do This:
Profile a 3rd party application or system application installed as an `.apk` file	1. Set the system property `dalvik.vm.dex2oat-flags` to `--generate-debug-info`: `adb shell setprop dalvik.vm.dex2oat-flags --generate-debug-info` 2. If you use `--compiler-filter=interpret-only`, set the optimization level to `speed`: `adb shell setprop dalvik.vm.dex2oat-filter speed` 3. (Re-)install the application. `adb shell install -r TheApp.apk`
Profile all applications installed as `.apk` or `.jar` files by	1. On your host system, open the `/build/core/dex_preopt_libart.mk` file, located in your Android OS directory structure. 2. Modify the `--no-generate-debug-info` line to `--generate-debug-info` and save and close the file.

To Do This:	Do This:
re-building the Android image when pre-optimization for private applications is enabled (`LOCAL_DEX_PREOPT:=true` property set in `device.mk`)	**3.** Rebuild the Android image and flash it to your device. **4.** If you are using an Android image that is not PIC configured (`WITH_DEXPREOPT_PIC:=false` property set in `device.mk`), generate `classes.dex` from odex using the `patchoat` command. `classes.dex` should appear in `/data/dalvik-cache/x86/system@app@appname@appname.apk@classes.dex`
Profile all applications installed as `.apk` or `.jar` files by re-building the Android image when pre-optimization for private applications is disabled (`LOCAL_DEX_PREOPT:=false` property set in `device.mk`)	**1.** Set the system property `dalvik.vm.dex2oat-flags` to `--generate-debug-info`: `adb shell rm -rf /data/dalvik-cache/x86/system@app@webview@webview.apk@classes.dex` `adb shell setprop dalvik.vm.dex2oat-flags --generate-debug-info` **2.** Stop and start the adb shell: `adb shell stop` `adb shell start` **3.** Generate the dex file: `adb shell ls /data/dalvik-cache/x86/system@app@webview@webview.apk@classes.dex` `adb pull /data/dalvik-cache/x86/system@app@webview@webview.apk@classes.dex`
Profile an application executed by the `dalvikvm` executable	Add the compiler option `--generate-debug-info` followed by `-Xcompiler-option`. Make sure the application has not been compiled yet. `rm -f /data/dalvik-cache/*/*TheApp.jar*` `adb shell dalvikvm -Xcompiler-option --include-debug-symbols -cp TheApp.jar`
Profile system and core classes	Set the system property `dalvik.vm.image-dex2oat-flags` to `--generate-debug-info` and force recompilation: `adb shell stop` `adb shell rm -f /data/dalvik-cache/*/*` `adb shell setprop dalvik.vm.dex2oat-flags --generate-debug-info` `adb shell setprop dalvik.vm.image-dex2oat-flags --generate-debug-info` `adb shell start` If you run the application before the system classes are compiled, you should add another compiler option `-Ximage-compiler-option --generate-debug-info`: `adb shell rm -f /data/dalvik-cache/*/*` `adb shell dalvikvm -Xcompiler-option --generate-debug-info -Ximage-compiler-option --generate-debug-info -cp TheApp.jar`

To Do This:	Do This:
NOTE This action is required if Java core classes get compiled to the `/data/dalvik-cache/` subdirectory. Manufacturers may place them in different directories. If manufactures supply the precompiled `boot.oat` file in `/system/framework/x86`, Java core classes will not be resolved because they cannot be re-compiled with debug information.	

Prepare an Android* Application for Analysis

Before starting an analysis with the VTune Amplifier, make sure your Android application is compiled with required settings:

Compilation Settings

Performance analysis is only useful on binaries that have been optimized and have symbols to attribute samples to source code. To achieve that:

- Compile your code with release level settings (for example, do not use the `/O0` setting on GCC*).
- Do not set `APP_OPTIM` to `debug` in your `Application.mk` as this setting disables optimization (it uses `/O0`) when the compiler builds your binary.
- To run performance analysis (Hotspots) on non-rooted devices, make sure to compile your code setting the `debuggable` attribute to `true` in `AndroidManifest.xml`.

> **NOTE**
> If your application is debuggable (`android:debuggable="true"`), the default setting will be `debug` instead of `release`. Make sure to override this by setting `APP_OPTIM` to `release`.

By default, the Android NDK build process for Android applications using JNI creates a version of your `.so` files with symbols.

The binaries with symbols included go to `[ApplicationProjectDir]/obj/local/x86`.

The stripped binaries installed on the target Android system via the `.apk` file go to `[ApplicationProjectDir]/libs/x86` . These versions of the binaries cannot be used to find source in the VTune Amplifier. However, you may collect data on the target system with these stripped binaries and then later use the binaries with symbols to do analysis (as long as it is an exact match).

When the VTune Amplifier finishes collecting the data, it copies `.so` files from the device (which have had their symbols stripped). This allows the very basic functionality of associating samples to assembly code.

Tip
Use ITT APIs to control performance data collection by adding basic instrumentation to your application.

Search Directories for Android* Targets

For accurate module resolution and source analysis of your Android* application, make sure to specify search paths for binary and source files when configuring performance analysis:

- from command line, use the `--search-dir`/`--source-search-dir` options; for example:

```
host>./amplxe-cl --collect hotspots -knob sampling-mode=hw -r system_wide_r@@@ --search-dir ~/
AndroidOS_repo/out/target/product/ctp_pr1/symbols/
```

- from GUI, use the Binary/Symbol Search and Source Search dialog boxes

If you have not set the project search directories at the time of collection or import, you will not be able to open the source code. Only Assembly view will be available for source analysis.

Consider the following when adding search paths:

- By default, the VTune Amplifier pulls many binaries from the target device.
- The Kernel `[vmlinux]` is one file that does not contain symbols on the target device. Typically it is located in `[AndroidOSBuildDir]/out/target/product/[your target]/linux/kernel/vmlinux`.
- Many operating system binaries with symbols are located in either `[AndroidOSBuildDir]/out/target/product/[your target]/symbols`, or `[AndroidOSBuildDir]/out/target/product/[your target]/obj`.
- Application binaries with symbols are located in `[AndroidAppBuildDir]/obj/local/x86`.
- Application source files for the C/C++ modules are usually located in `[AndroidAppBuildDir]/jni` , not in `[AndroidAppBuildDir]/src` (where the Java *source files are). Some third-party software in Android does not provide binaries with symbols. You must contact the third party to get a version of the binaries with symbols.
- You can see if a binary has symbols by using the `file` command in Linux and make sure that it says `not stripped`.

```
 file MyBinary.ext
MyBinary.ext: ELF 32-bit LSB shared object, Intel 80386, version 1
(SYSV), dynamically linked, not stripped
```

Targets in Virtualized Environments

Configure your system to use the Intel® VTune™ Amplifier for targets running in such virtualization environments as Hyper-V on Windows*, KVM* or Xen* on Linux*, Parallels* on macOS*, and others.*

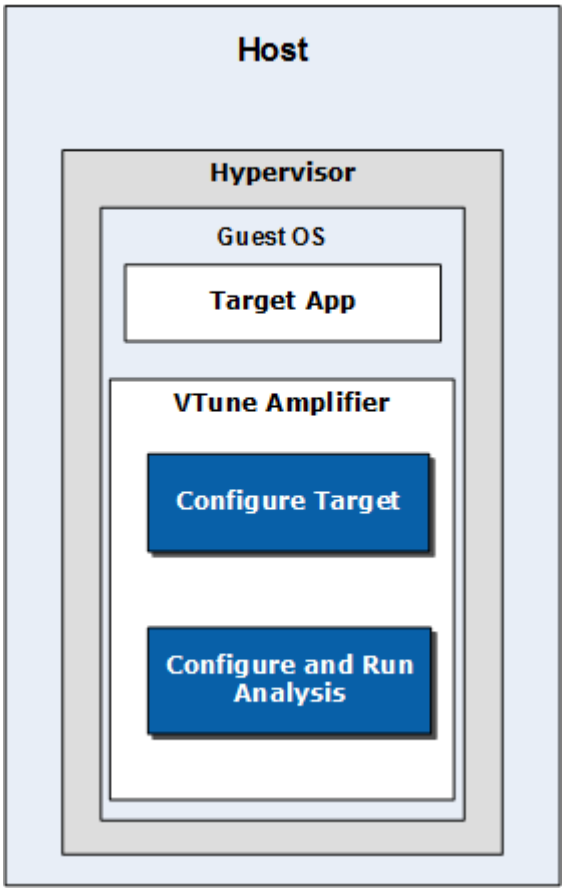
Virtual machines are made up of the following components:

- Host operating system: system from which the virtual machine is accessed. Supported host systems: Linux*, macOS*, Windows*
- Virtual machine manager (VMM) or cloud service provider: tool used to access and manage the virtual machine. Examples: VMware*, Parallels*

- Guest operating system: system accessed via the VMM and profiled using Intel VTune Amplifier. Supported guest systems: Linux*, Windows*

In most cases, the VTune Amplifier is installed on the guest operating system and analysis is run on the guest system. The guest system may not have full access to the system hardware to collect performance data. Analysis types that require access to system hardware, such as those that require precise event counters, will not work on a virtual machine.

NOTE
Typically the host operating system has access to the system hardware to collect performance data, but there are cases in which the host system may also be virtualized. If this is the case and you want to collect performance data on the host system, treat the host system as you would a guest system and assume that it no longer has the same level of access to the system hardware.



Virtual Machine Host/Guest Support

A typical virtualized environment includes a host operating system, which boots first and from which the VMM is loaded, and virtual machines (VMs) running guest operating systems. There are multiple combinations of each and support varies based on each component. For example, the VMM Parallels can be used only with a macOS host system, but can analyze performance with VTune Amplifier installed on either a Linux or Windows guest system.

	Linux Host	**Windows Host**	**macOS Host**
Linux Guest	KVM XEN Project	VMware	Parallels

	VMware		
Windows Guest	VMware	VMware Hyper-V	Parallels
macOS Guest	--	--	--

Additional steps are required to enable performance analysis on the virtual machine after Intel VTune Amplifier is installed. For more information about installing VTune Amplifier on a virtual machine, see the VTune Amplifier installation guide for your host operating system. Refer to the following topics to enable the vPMU for your hypervisor:

- Enable VMware analysis
- Enable KVM/XEN Project analysis

Analysis Type Support

Support for VTune Amplifier analysis types varies depending upon which counters have been virtualized by the VMM. If you run an analysis type that cannot be run in a virtualized environment, VTune Amplifier displays a warning message.

In general, the Hotspots analysis type in the user-mode sampling mode will work on every supported VMM because the analysis type does not require access to the system hardware. The Hotspots analysis in the hardware event-based sampling mode and other analysis types that use hardware event-based sampling collection have limited reporting functionality. For example, it will not include accurate results for stacks or call counts because this data relies on information provided by precise events (uncore). Running analysis types that rely on precise events will return results, but the collected data will be incomplete or distorted. That is, the result may not point to the actual instruction that caused the event, which can be difficult to differentiate from correct events. Refer to the documentation for your VMM to understand which counters have been virtualized.

- **Full Features**
  - Hotspots (user-mode sampling mode)
- **Limited Features**
  - Hotspots (hardware event-based sampling mode: Stacks not supported)
  - Microarchitecture Exploration: Stacks not supported

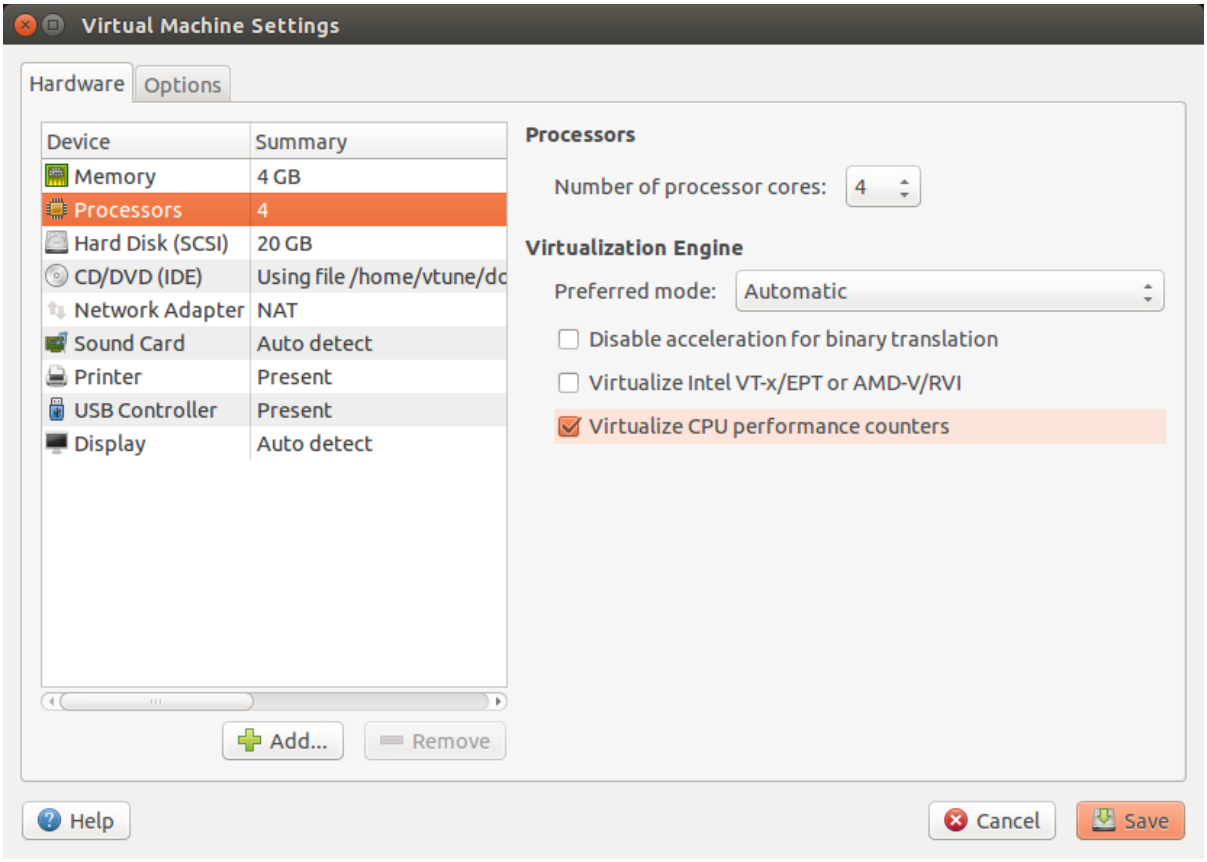
Profile Targets on a VMware* Guest System

Configure the Intel® VTune™ Amplifier to analyze performance on a VMware guest system.*

VMware users can use the VTune Amplifier to analyze a Windows* or Linux* virtual guest system. VTune Amplifier is installed and run on the guest system. Additional information about installing VTune Amplifier is available from the installation guides. Refer to the installation guide for the guest system operating system (Windows or Linux).

Use the following steps to enable event-based sampling analysis on the VMware virtual machine. Refer to the VMware documentation for the most up-to-date information.

1. From the host system, open the configuration settings for the virtual machine.
2. Select the **Processors** device on the left.
3. Select the **Virtualize CPU performance counters** checkbox.
4. Click **Save** to apply the change.



Profile Targets on a KVM* Guest System

Configure the Intel® VTune™ Amplifier to analyze performance on a KVM guest system.

Performance analysis for the host and virtual machine(s) in cloud environments helps identify such issues as resource contention (for example, CPU/vCPU time) and network/IO activity. VTune Amplifier uses Perf*-based driverless collection to enable performance analysis of the guest Linux* operating system via Kernel-based Virtual Machine (KVM) from the host system.

Unlike other virtual machine systems, systems using KVM on a Linux* host to access a Linux guest can have VTune Amplifier installed on either the host system to analyze performance on the guest system or installed directly on the guest system to analyze the guest system. Additional information about installing VTune Amplifier is available from the Linux installation guides.

Depending on your analysis target, you may choose any of the supported usage modes for KVM guest OS profiling.

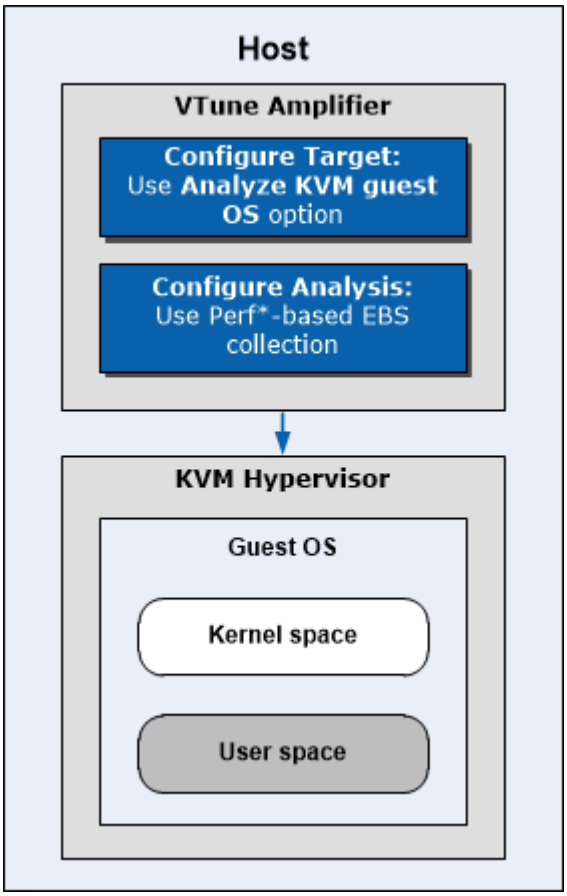
Profiling Modes

Currently, the VTune Amplifier supports the following usage modes for KVM guest OS profiling, and each of them has some limitations:

Profiling System	KVM Guest OS (User Apps)	KVM Guest OS (User and Kernel Space)	Host and KVM Guest OS (User and Kernel Space)	Host and KVM Guest OS (Kernel Space)
Supported analysis	User-mode sampling: Hotspots and Threading	Event-based sampling: Hotspots and limited Microarchitecture Exploration	Event-based sampling: all types with accurate attribution of user-space activity to the user processes on the guest	All Perf-based analysis types supported for the host and hotspots for the KVM kernel space
Target type	Applications in the Launch and Attach modes	• Applications in the Launch and Attach modes • System-wide analysis	System-wide analysis (host and guest OS)	All types supported for the host. No user app analysis on the guest OS.
VTune Amplifier installation mode	On the guest OS	On the guest OS	On the host and guest OS (VTune Amplifier custom collector)	On the host
Limitations	• Limited to Hotspots and Threading analysis • No system-wide analysis for user-mode sampling	• Limited event-based sampling analysis due to a limited set of virtualized PMU events and unavailable uncore events • No information from the host	• Additional debugfs and custom collector configuration is required • Access to the host system running VM is required • Not applicable to cloud environments	Guest OS analysis is limited: • Performance data is attributed to the kernel space only • No user-space analysis
Configuration	Learn more	PMU event virtualization required for Event-based sampling Learn more	**Analyze KVM guest OS** option Learn more	**Analyze KVM guest OS** option Learn more

Profile KVM Kernel Modules from the Host

If you are a system developer and interested in the performance analysis of a guest Linux* system including KVM modules, consider using this usage mode:



1. Prepare your system for analysis:
   a. Copy the `/proc/kallsyms` and `/proc/modules` files from a guest OS to a host file system to have KVM guest OS symbols resolved.
   b. Copy any guest OS's modules of interests (`vmlinux` and any `*.ko` files) from a guest OS and save them to a `[guest]` folder on the host file system.
2. Click the **Configure Analysis** button on the VTune Amplifier toolbar.

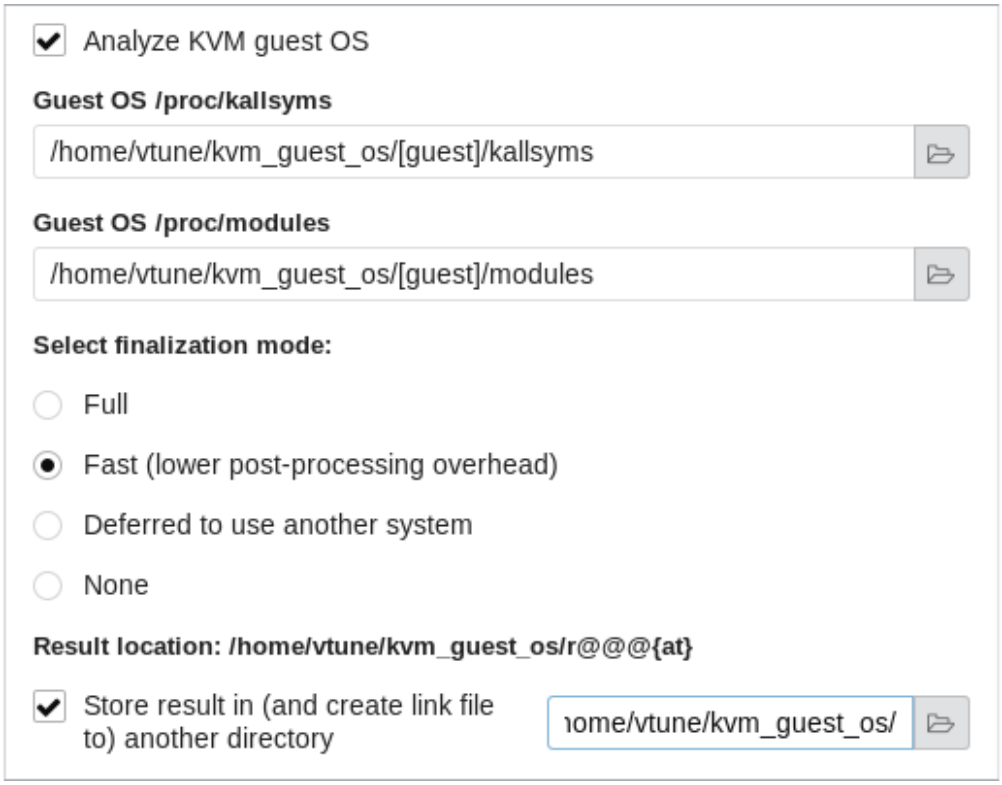
   The **Configure Analysis** window opens.
3. Make sure to select the **Local Host** target system in the **WHERE** pane and configure the required target type in the **WHAT** pane.

   By default, the **Launch Application** target type is selected.

   If you select the **Attach to Process** target type, specify the `qemu-kvm` process to attach to.

   Alternatively, you may specify the PID of the `qemu-kvm` process. To determine the PID, enter:

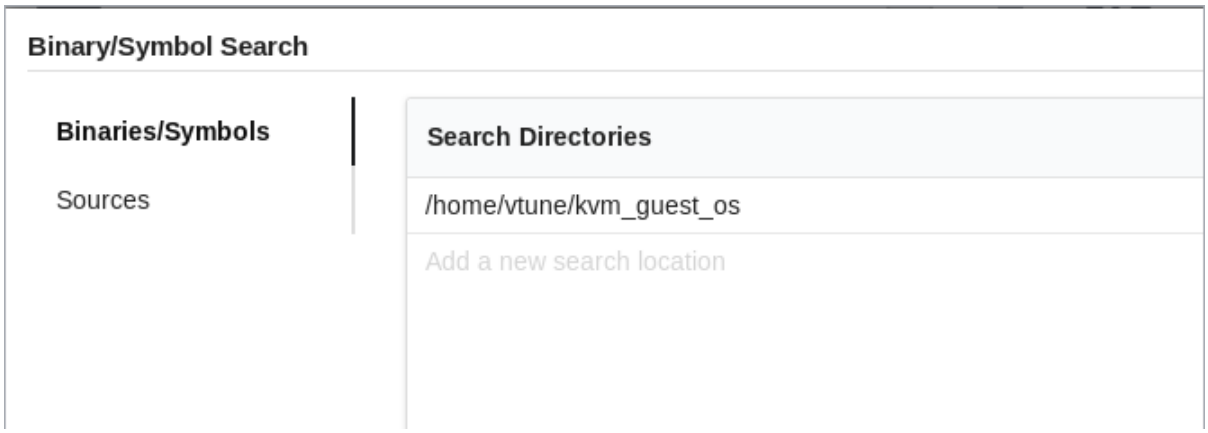
   ```
   $ ps aux | grep kvm
   ```
4. In the **Advanced** section of the **WHAT** pane, select the **Analyze KVM guest OS** option and enter paths to the local copies of the guest `/proc/kallsyms` and `/proc/modules` files; for example:



5. Click the **Search Binaries** button on the bottom right.

   The **Binary/Symbol Search** dialog box opens.
6. Add a local path to a `[guest]` folder where all modules copied from the guest OS reside.

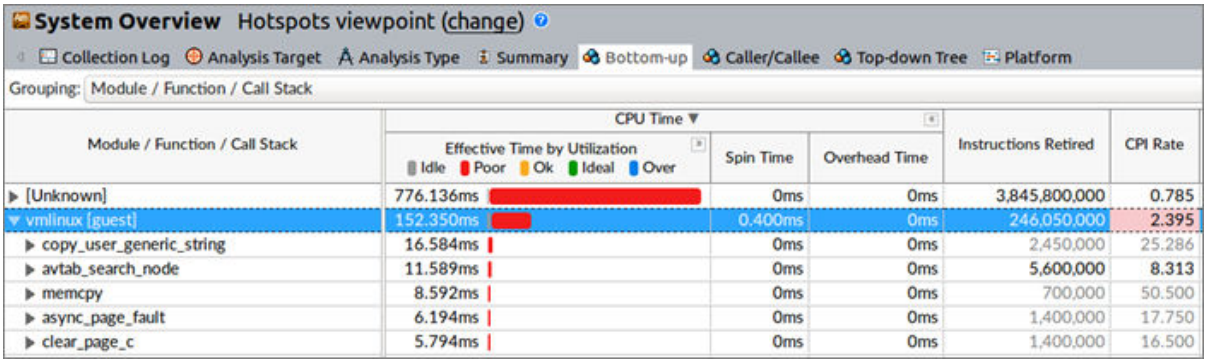
   For example, if your `[guest]` folder is located in `/home/vtune`, specify `/home/vtune` as a search directory:



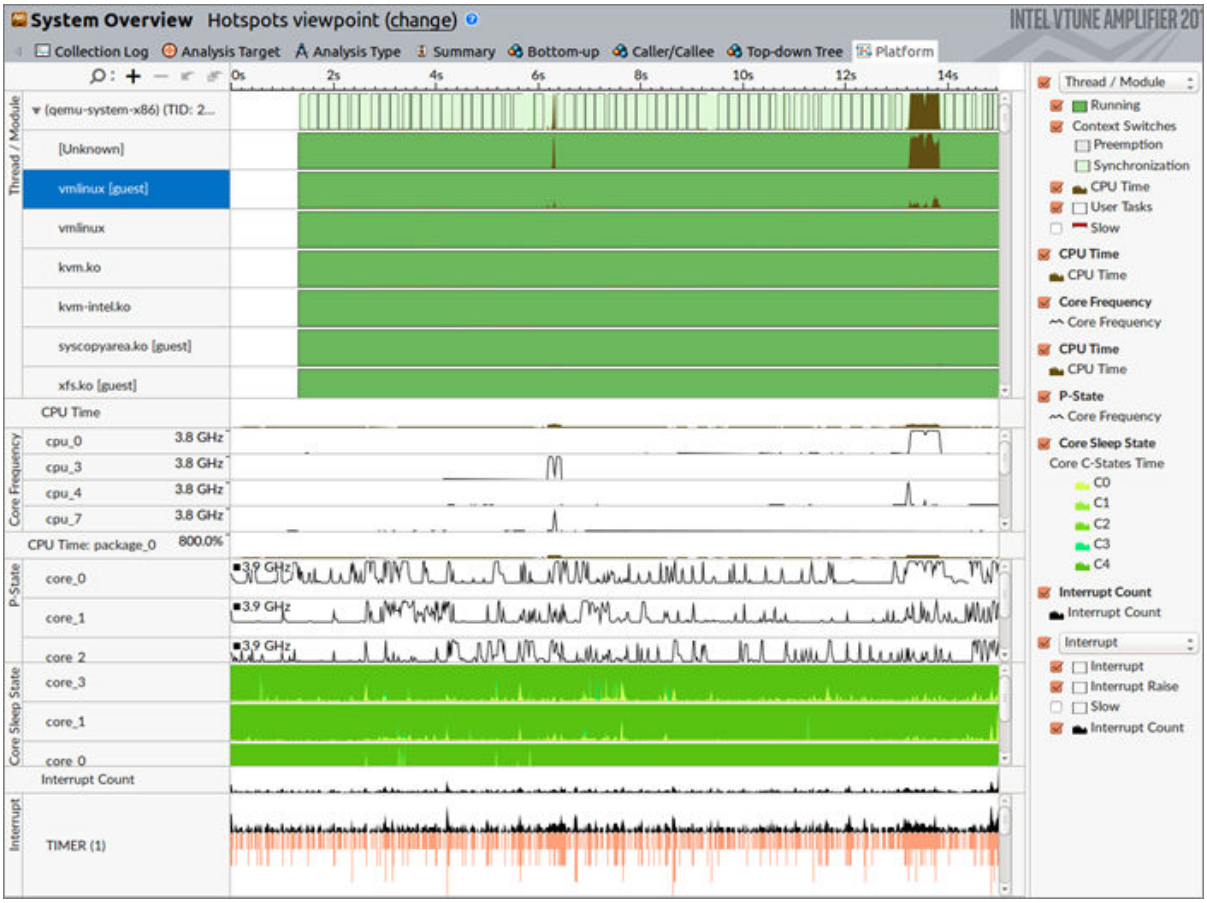
7. Click **OK** to save your changes.
8. In the **HOW** pane, select a required analysis type.

   For KVM guest OS profiling, you may choose analysis types using Perf*-based EBS data collection: Hotspots (hardware event-based sampling mode), System Overview, or configure your own custom analysis.
9. Click the **Start** button at the bottom to run the analysis.

When you run the analysis, the VTune Amplifier collects the data on both host and guest OS and displays merged statistics in the result. Guest OS modules have the `[guest]` postfix in the grid. For example:



Focus on the **Platform** tab to analyze your code performance on the guest OS and correlate this data with CPU, GPU, power, hardware event metrics and interrupt count at each moment of time. If you enabled the kvm Ftrace event collection for your target, you can also monitor the statistics for KVM kernel module:



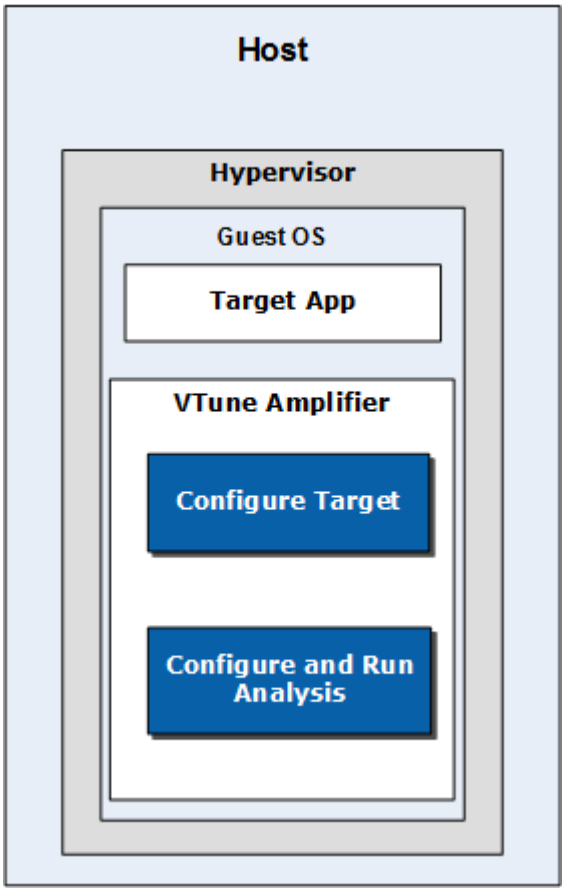
Limitations

- In this mode, the VTune Amplifier collects data only on the kernel space modules on the KVM guest OS. Data on user space modules shows up in the `[Unknown]` node and includes only high-level statistics.
- Call stack data is not collected for this type of profiling.

Profile KVM Kernel and User Space on the KVM System

Install the VTune Amplifier on the KVM system and configure your target for the KVM guest OS profiling.

For application analysis, you need to install the Intel® VTune™ Amplifier directly on your guest OS. VTune Amplifier installation detects a virtual environment and disables sampling drivers installation to avoid system instability. When the product is installed, proceed with project configuration by specifying your application as an analysis target and selecting an analysis type:



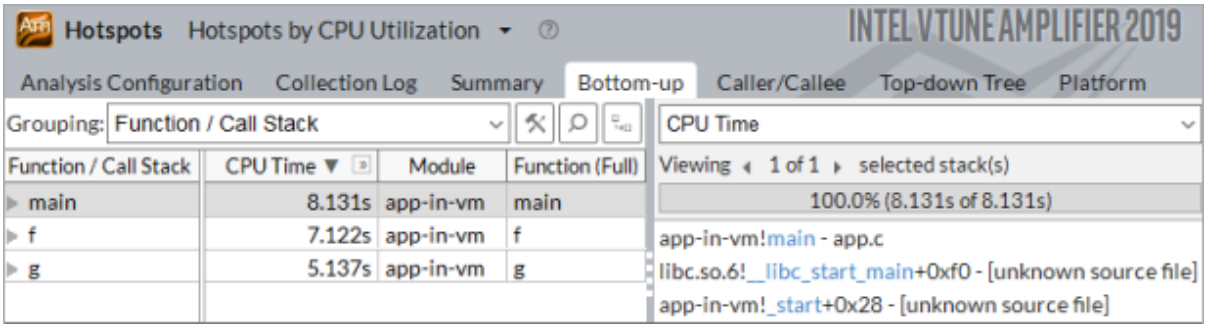
This profiling type supports two usage modes:

- Guest OS (user apps)
- Guest OS (kernel and user space)

Both profiling modes are applicable to cloud environments but introduce some limitations.

Guest OS (User App) Profiling Mode

In this mode, the VTune Amplifier supports user-mode sampling and tracing analysis types, Hotspots and Threading, for the applications running in the Launch or Attach mode. System-wide analysis is not supported.

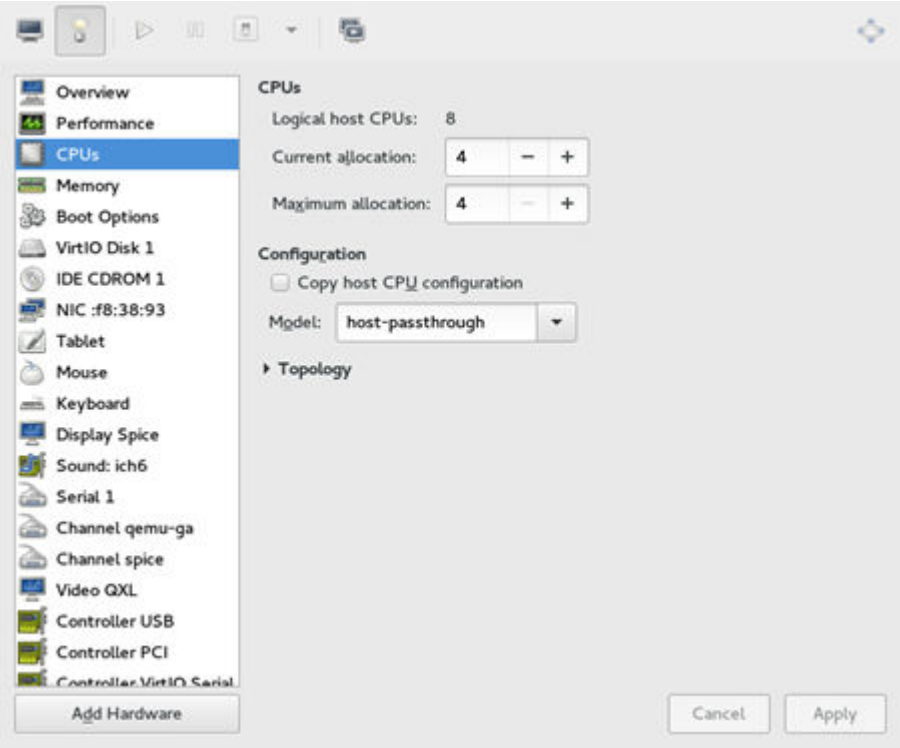


Guest OS (Kernel and User Space) Profiling Mode

In this mode, the VTune Amplifier provides limited event-based collection options for the Hotspots and Microarchitecture Exploration analyses and requires additional host system configuration to virtualize PMU counters.

To enable event-based sampling analysis on the KVM system:

1. From the host system, open the configuration settings for the virtual machine.
2. Select the **CPUs** or **Processor** option on the left.
3. Enter `host-passthrough` into the **Model** field to pass through the host CPU features without modifying the guest system.
4. Click **Apply** to save the changes.



When you select a hardware event-based analysis type (for example, Microarchitecture Exploration), the VTune Amplifier automatically enables a driverless event-based sampling collection using the Linux Perf* tool. For this analysis, the VTune Amplifier collects only architectural events. See the Performance Monitoring Unit Sharing Guide for more details on the supported architectural events.

Limitations

- User-mode sampling limitations:
  - Only Hotspots and Threading analyses are supported.
  - No system-wide analysis is available.
- Hardware event-based sampling limitations:
  - Only Hotspots and limited Microarchitecture Exploration analyses are supported.
  - PEBS counters are not virtualized.
  - Uncore events are not available.
- KVM modules and host system modules do not show up in the analysis result.
- Data on the guest OS and your application modules show up as locally collected statistics with no `[guest]` markers.

Profile KVM Kernel and User Space from the Host

In this mode, Intel® VTune™ Amplifier collects two traces in parallel: system-wide performance data trace on the host and OS-level event trace on the guest system. These traces get merged into one VTune Amplifier result and provide:

- simultaneous analysis of user space activity (processes, threads, functions) from the host on the guest system;
- accurate attribution of collected data to the user processes running on the guest, based on the timestamp synchronization.

This usage mode provides the following advantages:

- VMs are not required to virtualize performance counters. All performance analysis features are available to VM users out of the box.
- Sampling drivers (VTune Amplifier sampling driver or Perf*) do not need to be installed on a guest VM.

To enable KVM kernel and user space profiling from the host:

1. Install the VTune Amplifier on the host and virtual machine.

   NOTE
   You do not need to install sampling drivers.

2. On both host and guest systems, run the script from the bin64 folder as a root:

   ```
   $ prepare-debugfs.sh -g <user_group>
   ```

   ```
   $ echo 0 > /proc/sys/kernel/perf_event_paranoid
   ```
3. Configure a password-less SSH access from the host to the KVM guest system.
4. If your host system is multi-socket, export the environment variable to set the time source to TSC before starting the VTune Amplfier:
   `AMPLXE_RUNTOOL_OPTIONS=--time-source=tsc`
5.
6. From the **WHAT** pane in the **Configure Analysis** window, expand the **Advanced** section and enter the following string to the **Custom collector** field:
   `python <vtune_install_dir>/bin64/kvm-custom-collector.py --kvm-ssh-login=<username>@<kvm_ssh_ip> --vtune-dir-on-kvm=<vtune-install-dir>`

   NOTE
   For additional details on particular options, see the `kvm-custom-collector.py` script help.

7. To collect data from the guest kernel space, select the **Analyze KVM Guest OS** option.

   Copy `/proc/kallsyms` and `/proc/modules` files from the virtual machine to the host.

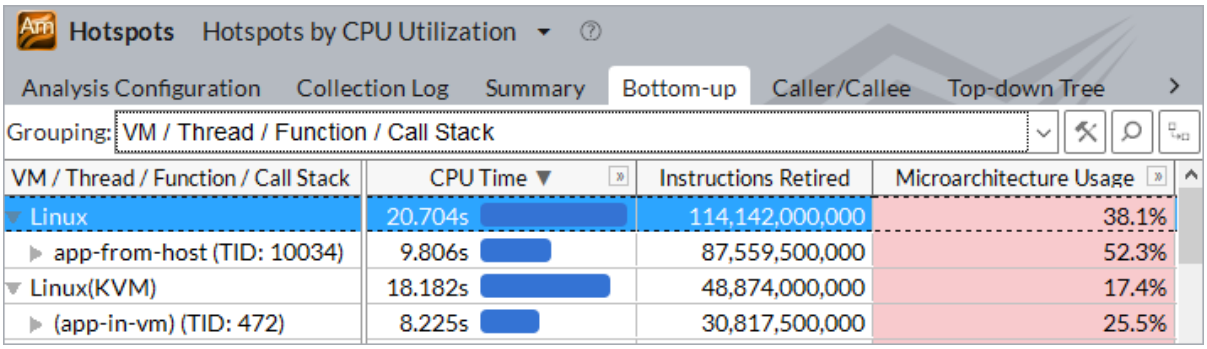
   NOTE
   Since these are pseudo-files, you are recommended to `cat` their content into a regular file and then copy it to the host. Specify paths to the copied files in the project properties.

8. From the **HOW** pane, select any hardware event-based sampling analysis (for example, General Exploration) and run the analysis from the host.

Explore the collected data by enabling all the grouping levels containing a VM component to differentiate the host and target data.

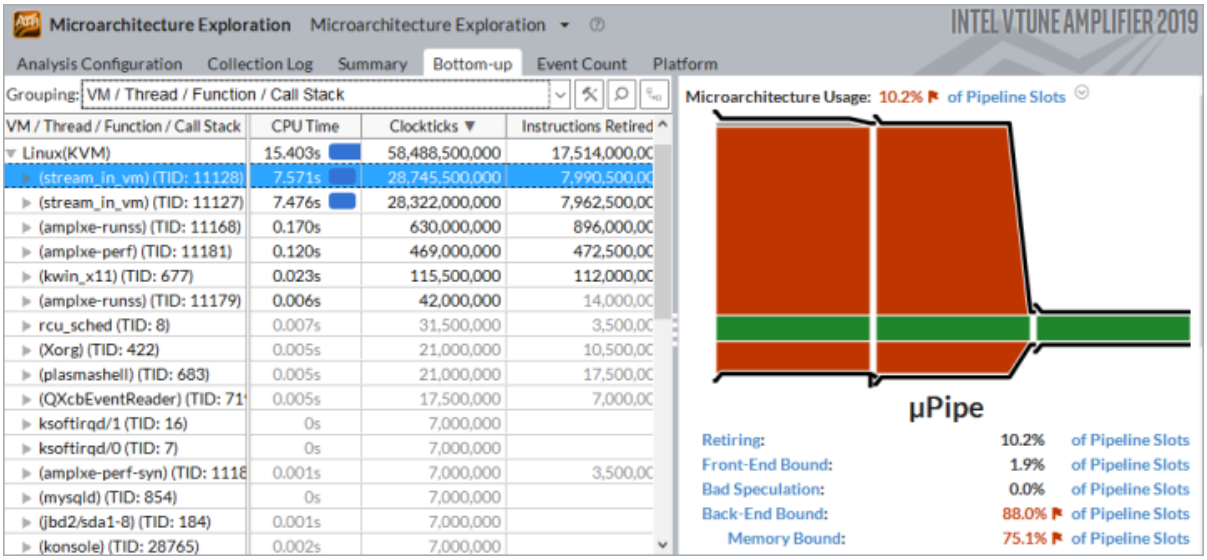
Example 1: Hotspots Analysis (Hardware Event-Based Sampling Mode)

Analyze hotspots for both an application launched from the Linux host, `app-from-host`, and an application launched on the KVM guest system, `app-in-vm`:



Example 2: Microarchitecture Exploration Analysis

Analyze the efficiency of the Microarchitecture Usage for the application launched on the KVM guest system. The context summary on the right pane shows the hardware metrics for the thread (launched inside the KVM) selected in the grid:



System Requirements and Limitations

- Minimum Linux kernel version for host system is 4.9.
- debugfs is mounted on both host and guest system.
- Irrespective of the number of KVM/Qemu processes running, only one running VM instance can be profiled.
- In the result view, threads with the same name may be grouped into one process (`ftrace`).
- In the result view, samples before the first context switch may be attributed to the hypervisor thread on the host.

Profile Targets on a Xen* Virtualization Platform

Configure Intel® VTune™ Amplifier and your system with a Xen virtualization platform for performance profiling.

You can use the VTune Amplifier for hardware event-based analysis either for a guest OS (DomU), a privileged OS (Dom0), or all the domains at once.

Configure a Target System for Analysis

Before running a VTune Amplifier analysis on a system with a Xen virtualization platform, enable full-platform CPU monitoring required for event-based sampling analysis:

```
$ echo "all" > /sys/hypervisor/pmu/pmu_mode
```

To get CPU profiling data on a virtualized system (Dom0 and the hypervisor only), enter:

```
$ echo "hv" > /sys/hypervisor/pmu/pmu_mode
```

NOTE

- Some configurations do not support the `all` mode.
- CPU events virtualization requires root privileges.
- Unlike CPU profiling, GPU profiling in the `hv` mode is available for all domains (Dom0 and DomU).

Configure VTune Amplifier for Xen Platform-Wide Analysis

Prerequisites: Make sure the Dom0 remote analysis target is accessible via the Ethernet/SSH connection from your host without any password.

Create a VTune Amplifier project and specify options for your remote target as follows:

1. Select the **remote Linux (SSH)** type of the target system on the **WHERE** pane.
2. Specify **SSH destination** details for your Dom0 remote target system.
3. Select the **Profile System** target type to enable platform-wide performance monitoring (**WHAT** pane).

   As soon as you set up the target options, the VTune Amplifier attempts to automatically install required components on the specified remote system. If, for some reason, the system cannot be reached, VTune Amplifier displays an error message. To troubleshoot this potential problem, make sure the default path specified as the **VTune Amplifier installation on the remote system** in the **WHERE** pane is accessible, writable, and has 200Mb of available space. If not, specify another location, for example: `/tmp`.

As soon as the connection is established and the target is configured, select an analysis type supported on the Xen virtualization platform from the **HOW** pane:

- Microarchitecture Exploration
- GrPU Rendering(preview)

Targets in a Cloud Environment

You can use the Intel® VTune™ Amplifier to run application performance analysis in the user-mode sampling mode on Windows* or Linux* virtual machine based instances or any analysis type on a bare-metal cloud instance.

The follow cloud service providers are supported:

- Amazon Web Services* (AWS)
- Google Cloud Platform*
- Microsoft Azure*

You can install VTune Amplifier either directly on the cloud instance or on a Windows, Linux, or macOS* host system and target a Linux cloud instance for remote analysis.

Prerequisites:

- Existing account with one of the supported cloud service providers
- Existing Linux or Windows instance in the cloud

- Linux instance: Root or sudo privileges to enable user-mode sampling Hotspots analysis by setting `/proc/sys/kernel/yama/ptrace_scope` to `0`. See the *Intel VTune Amplifier Release Notes* for instructions on enabling it permanently.
- If installing in the cloud: At least 25GB of instance storage

To install VTune Amplifier on the cloud instance, copy the VTune Amplifier installer to the cloud instance and run the installer.

A use case with steps for installing and configuring VTune Amplifier on an Amazon Web Services instance and running a Hotspots analysis on that instance is available from https://software.intel.com/en-us/vtune-amplifier-cookbook-profiling-applications-in-aws-ec2-instances.

Configure SSH Access for Remote Collection

To collect data on a remote Linux system, configure SSH to work in a password-less mode so it does not prompt for the password on each invocation.*

NOTE
A root connection is required to load the sampling drivers and to collect certain performance metrics. You can configure the system using root permissions and then set up password-less SSH access for a non-root user if desired. For example, build and load the sampling drivers on the target system using root access and then connect to the system and run analysis as a non-root user. If you set up access without using the sampling drivers, then driverless event-based sampling data can still be collected.

Use one of the methods below to enable password-less SSH access:

- Manually configure a connection from macOS*/Linux to Linux

Configure a Password-less SSH Access from Linux/macOS to Linux

For remote collection on a Linux target system, set up the password-less mode on the local Linux or macOS host as follows:

1. Generate the key with an empty passphrase:

   ```
   host> ssh-keygen -t rsa
   ```

2. Copy the key to target system:

   ```
   host> ssh-copy-id user@target
   ```

   Alternatively, if you do not have `ssh-copy-id` on your host system, use the following command:

   ```
   host> cat .ssh/id_rsa.pub | ssh user@target 'cat >> .ssh/authorized_keys'
   ```

3. Verify that a password is not required anymore, for example:

   ```
   host> ssh user@target ls
   ```

Configure system for GPU Application Analysis on Intel® HD Graphics and Intel® Iris® Graphics

To analyze Intel HD and Intel Iris Graphics (further: Intel Graphics) hardware events on a GPU, your system needs to have Intel Metric Discovery API library installed. Depending on your target analysis system, do the following:

- **Windows* systems:** Intel Metric Discovery API library is part of the official Intel Graphics driver package. You can install a driver for your system from https://downloadcenter.intel.com.
- **Linux* systems:** installation of the Intel Metric Discovery API library depends on your system and kernel version.

  The Get Started Guide is available from https://software.intel.com/en-us/articles/sdk-for-opencl-gsg.

- For CentOS* 7, make sure to install a proper version of the Intel Media Server Studio (starting with version 2015 R5) and build the kernel driver as described in the Get Started Guide.
- For other Linux systems, check the kernel version:

4.14 and higher	To analyze GPU usage, configure your kernel as follows: `CONFIG_EXPERT=y`, `CONFIG_DRM_I915_LOW_LEVEL_TRACEPOINTS=y`. If the VTune Amplifier cannot collect GPU hardware metrics and provides a corresponding error message, make sure you have installed Metrics Discovery API library from https://github.com/intel/metrics-discovery correctly. See the Troubleshooting topic for details.
4.4 or 4.7	Install Intel OpenCL SDK to patch and re-build the kernel as described in the Get Started Guide.
4.4.111 4.4 3.14.5	Install Intel Media Server Studio version applicable to your kernel version: • Intel Media Server Studio 2018 for 4.4.111 • Intel Media Server Studio 2017 for 4.4 • Intel Media Server Studio 2016 for 3.14.5 Build the kernel driver as described in the *Generic* section of the Get Started Guide.

Rebuild and Install the Kernel for GPU Analysis

To enable GPU usage events (i915 ftrace events) collection, your Linux kernel should be properly configured. For example, for kernel 4.14 and higher, these settings should be enabled: `CONFIG_EXPERT=y` and `CONFIG_DRM_I915_LOW_LEVEL_TRACEPOINTS=y`.

If the Intel® VTune™ Amplifier cannot start an analysis and provides an error message "*Collection of GPU usage events cannot be enabled. i915 ftrace events are not available*", you need to rebuild and install the re-configured kernel. For example, for Ubuntu* 16.04.4 (kernel 4.14.20) running on Intel microarchitecture code name Skylake, configure and install the kernel as follows:

NOTE
Configuring kernel requires root permissions.

1. Add (or uncomment) source package repositories for your Ubuntu version in `/etc/apt/sources.list`.

   For example, on Ubuntu Xenial Xerus* you should have:

```
deb-src http://archive.ubuntu.com/ubuntu xenial main
deb-src http://archive.ubuntu.com/ubuntu xenial-updates main
```

2. Update your system to prepare it for building the kernel:

```
sudo apt update
sudo apt build-dep linux-image-`uname -r`
sudo apt install fakeroot libssl-dev
```

3. Install kernel sources.

   If you have a standard kernel coming with your distribution, you can retrieve kernel sources with your package manager:

```
sudo apt install linux-source
```

   The kernel source is installed to `/usr/src/linux-source-x.y.z/linux-source-x.y.z.tar.bz2`. Extract the kernel source from the archive and use this source directory.

   If you have a custom kernel, you need to find the corresponding source code where the kernel belongs.

4. Create a `.config` file with the same configuration you have for your running kernel:

```
cp /boot/config-`uname -r` .config
make olddefconfig
```

5. In the new `.config` file, make sure the following settings are enabled:

   ```
   CONFIG_EXPERT=y
   CONFIG_FTRACE=y
   CONFIG_DEBUG_FS=y
   CONFIG_DRM_I915_LOW_LEVEL_TRACEPOINTS=y
   ```

   Update the file, if required, and save.

6. Build `objtool`. This tool is required for building the sampling driver.

```
sudo apt install libelf-dev
make -C tools/ objtool
```

7. Build the kernel with the new `.config` file:

```
make -j `getconf _NPROCESSORS_ONLN` deb-pkg
```

8. Install the kernel and kernel modules:

```
sudo dpkg -i linux-*.deb
```

9. Reboot the machine with the new kernel.

Verify Intel® VTune™ Amplifier Installation on a Linux* System

A self-check script is available to validate that appropriate drivers are installed and the system is set up properly to collect performance data. The script can be run on individual systems or on a cluster environment.

The `amplxe-self-checker.sh` script is available from `<install-dir>/bin64` on the Linux system on which VTune Amplifier is installed. The script runs several representative analysis types on a sample with reliable hotspots. After the script completes, it produces a log file and gives diagnostics on the success or failure of the checks. The analysis types that are launched cover:

- Software sampling and tracing collection (Hotspots in the user-mode sampling mode or Threading)
- Core event-based sampling collection (Hotspots in the hardware event-based sampling mode with and without stacks)
- Microarchitecture Exploration analysis
- Memory Access analysis with uncore events

The diagnostics detect possible collection limitations and lists steps to overcome the limitations. The output also provides information about missing permissions or outdated drivers.

Use the `--log-dir` option when running the script to specify a location for the log file to be stored. This option is useful when running the script on a compute node through a job scheduler.

Installing Command Line Collectors

You can install the command line data collection features of the product on a system to reduce disk space footprint and simply collect data. Data collection does not require a license; however, viewing of the data cannot be done unless a license is present. The results of any data collection that is run on the system must then be copied to a system where the regular install was done for analysis, viewing, and reporting.

1. On the host system, extract the installation package to a writeable directory with the following command:

```
tar -xzf vtune_amplifier_<version>.tar.gz
```

2. Navigate to the directory containing the extracted files.
3. Copy the `CLI_install` folder (found at the top level in the extracted product install package) to the remote machine.
4. Execute `./install.sh` script file (this file is located inside the `CLI_install` folder). No activation is required.

Sampling Drivers

Intel® VTune™Amplifier uses kernel drivers to enable the hardware event-based sampling. VTune Amplifier installer automatically uses the Sampling Driver Kit to build drivers for your kernel with the default installation options. If the drivers were not built and set up during installation (for example, lack of privileges, missing kernel development RPM, and so on), VTune Amplifier provides an error message and, on Linux* and Android* systems, enables driverless sampling data collection based on the Linux Perf* tool functionality, which has some analysis limitations for a non-root user. VTune Amplifier also automatically uses the driverless mode on Linux when hardware event-based sampling collection is run with stack analysis, for example, for Hotspots or Threading analysis types.

If not used by default, you may still enable a driver-based sampling data collection by building/installing the sampling drivers for your target system:

- Linux* targets:
  - Make sure the driver is installed.
    - Build the driver, if required.
    - Install the driver, if required.
  - Verify the driver configuration.
- Android* targets: Verify the sampling driver is installed. If required, build and install the driver.

NOTE

- You may need kernel header sources and other additional software to build and load the kernel drivers on Linux. For details, see the `README.txt` file in the `sepdk/src` directory.
- A Linux kernel update can lead to incompatibility with VTune Amplifier drivers set up on the system for event-based sampling (EBS) analysis. If the system has installed VTune Amplifier boot scripts to load the drivers into the kernel each time the system is rebooted, the drivers will be automatically re-built by the boot scripts at system boot time. Kernel development sources required for driver rebuild should correspond to the Linux kernel update.
- If you loaded the drivers but do not use them and no collection is happening, there is no execution time overhead of having the drivers loaded. The memory overhead is also minimal. You can let the drivers be loaded at boot time (for example, via the `install-boot-script`, which is used by default) and not worry about it. Unless data is being collected by the VTune Amplifier, there will be no latency impact on system performance.

Build and Install the Sampling Drivers for Linux* Targets

Prerequisites for remote Linux target systems: You need root access to the target system.

Prerequisites for all Linux systems: Check the Intel® Developer Zone for updated versions of the sampling driver between releases of Intel® VTune™ Amplifier. Sampling driver downloads are available from https://software.intel.com/en-us/articles/intel-vtune-amplifier-sampling-driver-downloads.

Install Drivers on Linux* Host Systems

During product installation on a host Linux OS, you may control the drivers installation options via the Advanced Options. VTune Amplifier provides the following options:

Use This Option	To Do This
Sampling driver install type [build driver (default) / driver kit files only]	Choose the driver installation option. By default, VTune Amplifier uses the Sampling Driver Kit to build the driver for your kernel. You may change the option to **driver kit files only** if you want to build the driver manually after installation.
Driver access group [vtune (default)]	Set the driver access group ownership to determine which set of users can perform the collection on the system. By default, the group is `vtune`. Access to this group is not restricted. To restrict access, see the **Driver permissions** option below. You may set your own group during installation in the Advanced options or change it manually after installation by executing: `./boot-script --group <your_group>` from the `<install_dir>/sepdk/src` directory.
Driver permissions [660 (default)]	Change permissions for the driver. The default permissions allow any user to access the driver. Using this access the user can profile the system, an application, or attach to a process.
Load driver [yes (default)]	Load the driver into the kernel.
Install boot script [yes (default)]	Use a boot script that loads the driver into the kernel each time the system is rebooted. The boot script can be disabled later by executing: `./boot-script --uninstall` from the `<install_dir>/sepdk/src` directory.
Enable per-user collection mode [no (default) / yes]	Install the hardware event-based collector driver with the per-user filtering on. When the filtering is on, the collector gathers data only for the processes spawned by the user who started the collection. When it is off (default), samples from all processes on the system are collected. Consider using the filtering to isolate the collection from other users on a cluster for security reasons. The administrator/root can change the filtering mode by rebuilding/restarting the driver at any time. A regular user cannot change the mode after the product is installed. **NOTE** For MPI application analysis on a Linux* cluster, you may enable the **Per-user Hardware Event-based Sampling** mode when installing the Intel Parallel Studio XE Cluster Edition. This option ensures that during the collection the VTune Amplifier collects data only for the current user. Once enabled by the administrator during the installation, this mode cannot be turned off by a regular user, which is intentional to preclude individual users from observing the performance data over the whole node including activities of other users. After installation, you can use the respective `-vars.sh` files to set up the appropriate environment (PATH, MANPATH) in the current terminal session.

Use This Option	To Do This
Driver build options ...	Specify the location of the kernel header files on this system, the path and name of the C compiler to use for building the driver, the path and name of the make command to use for building the driver.

Check Sampling Driver Installation

To verify that the sampling driver is installed correctly on the host Linux system:

1. Check whether the sampling drivers are installed:

   ```
   $ cd <install-dir>/sepdk/src
   $ ./insmod-sep -q
   ```

   This provides information on whether the drivers are currently loaded and, if so, what the group ownership and file permissions are on the driver devices.
2. Check group permissions.

   If drivers are loaded, but you are not a member of the group listed in the `query` output, request your system administrator to add you to the group. By default, the driver access group is `vtune`. To check which groups you belong to, type `groups` at the command line. This is only required if the permissions are other than 660 or 666.

   NOTE
   If there is no collection in progress, there is no execution time overhead of having the driver loaded and very little overhead for memory usage. You can let the system module be automatically loaded at boot time (for example, via the `install-boot-script` script, used by default). Unless the data is being collected by the VTune Amplifier, there will be no latency impact on the system performance.

Verify Kernel Configuration

To verify kernel configuration:

1. Make sure that the kernel header sources are present on your host system. The kernel version should be 2.6.28 or later. To find the kernel version, explore `kernel-src-dir/include/linux/utsrelease.h`, or, depending on the kernel version: `kernel-src-dir/include/generated/utsrelease.h`. For more details, see the `README.txt` file in the `sepdk/src` directory.
2. Make sure the following options are enabled in the kernel configuration for hardware event-based sampling (EBS) collection:
   - `CONFIG_MODULES=y`
   - `CONFIG_MODULE_UNLOAD=y`
   - `CONFIG_PROFILING=y`
   - `CONFIG_SMP=y`
   - `CONFIG_TRACEPOINTS=y` (optional but recommended)
3. In addition to the options above, make sure the following options are enabled in the kernel configuration for EBS collection *with stacks*:
   - `CONFIG_KPROBES=y`
   - `CONFIG_RING_BUFFER=y`
   - `CONFIG_FRAME_POINTER=y` (optional but recommended for kernel stack analysis)
4. For remote target systems, determine if signed kernel modules are required (`CONFIG_MODULE_SIG_FORCE=y`). If they are, you must have the signed key that matches your target system.

If you are building the sampling drivers from a fresh kernel source and want to use it for an existing target system, get the original key files and sign the sampling driver with the original key. Alternatively, build the new kernel and flash it to the target device so the target device uses your kernel build.

Build the Sampling Driver

Prerequisites:

- You need kernel header sources and other additional software to build and load the kernel drivers on Linux. Refer to the Verify kernel configuration section.
- To cross-build drivers for a remote target Linux system, extract the package from the `<install_dir>/target` folder to `<extract_dir>`.

To build the driver if it is missing:

1. Change the directory to locate the build script:
   - To build drivers for a local system: `> cd <install_dir>/sepdk/src`
   - To cross-build drivers for a remote target system: `> cd <extract_dir>/sepdk/src`
2. Use the `build-driver` script to build the drivers for your kernel. For example:

- `$ ./build-driver`

  The script prompts the build option default for your local system.
- `$ ./build-driver -ni`

  The script builds the driver for your local system with default options without prompting for your input.
- `$ ./build-driver -ni -pu`

  The script builds the driver with the per-user event-based sampling collection on without prompting for your input.
- `$ ./build-driver -ni \`

  `--c-compiler=i586-i586-xxx-linux-gcc \`

  `--kernel-version="<kernel-version>" \`

  `--kernel-src-dir=<kernel-source-dir> \`

  `--make-args="PLATFORM=x32 ARITY=smp"`

  `--install-dir=<path>`

  The script builds the drivers with a specified cross-compiler for a specific kernel version. This is usually used for the cross-build for a remote target system on the current host. This example uses the following options:

  - `-ni` disables the interactive during the build.
  - `--c-compiler` specifies the cross build compiler. The compiler should be available from the PATH environment. If the option is not specified, the host GCC compiler is used for the build.
  - `--kernel-version` specifies the kernel version of the target system. It should match the `uname -r` output of your target system and the UTS_RELEASE in `kernel-src-dir/include/generated/utsrelease.h` or `kernel-src-dir/include/linux/utsrelease.h`, depending on your kernel version.
  - `--kernel-src-dir` specifies the kernel source directory.
  - `--make-args` specifies the build arguments. For a 32-bit target system, use `PLATFORM=x32`. For a 64-bit target system, use `PLATFORM=x32_64`
  - `--install-dir` specifies the path to a writable directory where the drivers and scripts are copied after the build succeeds.

Use `./build-driver -h` to get the detailed help message on the script usage.

To build the sampling driver as RPM using build services such as Open Build Service (OBS):

Use the `sepdk.spec` file located at the `<install_dir>/sepdk/src` directory.

Install the Sampling Drivers

Prerequisites for remote target systems: Copy the `sepdk/src` folder or the folder specified by the `--install-dir` option when building the driver to the target system using ssh, ftp, adb, sdb, or other supported means.

To install the drivers:

1. If building the drivers succeeds, install them manually with the `insmod-sep` script:

   ```
   $ cd <install_dir>/sepdk/src
   $ ./insmod-sep -r -g <group>
   ```

   where `<group>` is the group of users that have access to the driver.

   To install the driver that is built with the per-user event-based sampling collection on, use the `-pu (-per-user)` option as follows:

   ```
   $ ./insmod-sep -g <group> -pu
   ```

   If you are running on a resource-restricted environment, add the `-re` option as follows:

   ```
   $ ./insmod-sep -re
   ```

2. Enable the Linux system to automatically load the drivers at boot time:

   ```
   $ cd <install_dir>/sepdk/src
   $ ./boot-script --install -g <group>
   ```

   The `-g <group>` option is only required if you want to override the group specified when the driver was built.

To remove the driver on a Linux system, run:

```
./rmmod-sep -s
```

Build and Install Sampling Drivers for Android* Targets

On some versions of Android systems, including most of the Intel® supplied reference builds for SDVs, the required drivers are pre-installed in `/lib/modules` or `/system/lib/modules`. If the drivers are not pre-installed in any of these directories, you need to build them manually from the command line. Optionally, you can get the drivers integrated into the Android build so that they are built and installed when the operating system is built.

Android requires signed drivers. Every time the Android kernel is built, a random private/public key is generated. Drivers must be signed with the random private key to be loaded. The drivers (`socperf2_x.ko`, `pax.ko` , `sep4_x.ko` , and `vtsspp.ko`) must be signed with the same key and be compiled against the same kernel headers/sources as what is installed on the Android target system.

VTune Amplifier has options for building a new driver on the Linux host system and installing it on a target Android system. This is not the default and will only work if you provide the proper kernel headers/sources and a signing key. For example, the VTune Amplifier uses the `--with-drivers` option for building PMU drivers and `--kernel-src-dir` option for providing the configured kernel headers/sources tree path.

To build the sampling drivers on the host Linux system, enter:

```
<install-dir>/bin{32,64}/amplxe-androidreg.sh --package-command=build --with-drivers --kernel-src-dir=/ path /to/configured/kernel/sources [--jitvtuneinfo=jit|src|dex|none]
```

To install the sampling drivers from the Linux host, enter:

```
<install-dir>/bin{32,64}/amplxe-androidreg.sh --package-command=install --with-drivers
--kernel-src-dir=/ path/to/configured/kernel/sources [--jitvtuneinfo=jit|src|dex|none]
```

To sign the drivers after the drivers are built:

Typically the VTune Amplifier automatically signs drivers if kernel sources with the keys are available when it builds the drivers. Otherwise, to manually sign the drivers, use the following command:

```
$KERNEL_SRC/source/scripts/sign-file CONFIG_MODULE_SIG_HASH $KERNEL_SRC/
signing_key.priv $KERNEL_SRC/signing_key.x509 driver.ko
```

where the `CONFIG_MODULE_SIG_HASH` value is extracted from the `$KERNEL_SRC/.config` file.

> **NOTE**
> You need the "exact" signing key that was produced at the time and on the system where your kernel was built for your target.

Driverless Event-Based Sampling Collection

To enable hardware event-based sampling analysis on your platform, the Intel® VTune™ Amplifier can use either sampling drivers that require root privileges for installation on the Linux* and Android* systems or the Perf* utility, which is part of the default VTune Amplifier installation package.

VTune Amplifier runs the hardware event-based sampling analysis in the driverless mode with the Perf utility if:

- The sampling drivers cannot be installed (for example, if installed without root privileges)
- Collection with stacks is selected with a non-zero stack size and the prerequisites for driverless collection are satisfied
- The option to use driverless collection is selected in the VTune Amplifier user interface and the prerequisites for driverless collection are statisfied

> **NOTE**
> VTune Amplifier is installed to your default account. For non-root users, it provides a notification during the installation claiming that the sampling driver cannot be installed, so some product features could be limited or unavailable. To have the sampling driver installed, you need to re-start the install process under the root account or contact your administrator.

On Linux, the driverless collection is a default mode for analyses based on hardware event-based sampling with stacks (for example, Hotspots or Threading).

- Prerequisites
- Driverless collection modes
- Analysis limitations

Prerequisites for Driverless Collection

VTune Amplifier can use the driverless Perf-based collection if the following requirements are satisfied:

- Your system is based on kernel 2.6.32 or higher, which exports CPU PMU programming details over `/sys/bus/event_source/devices/cpu/format` file system.
- Perf-based collection is enabled in the kernel with a `/proc/sys/kernel/perf_event_paranoid` value equal to or less than 1.
- For uncore event analysis, `uncore_*` devices are available in the `/sys/bus/event_source/devices` folder.

- Context switch data cannot be collected using Perf-based driverless collection if the kernel version is less than 4.3.
- The types of context switches (preemption or synchronization) may not be identified on kernels older than 4.17.
- Hardware event-based sampling analysis is configured to collect stacks.

Driverless Collection Modes

VTune Amplifier supports the following Perf-based collection types:

- **Driverless Perf per-process sampling** collects samples for a single process and/or its children and can be done simultaneously from multiple monitoring processes. Since it requires performance counters virtualization per process, it can bring more overhead in comparison with system-wide collection. Typically, a system has this type of collection enabled by default.
- **Driverless Perf system-wide sampling** is performed by one monitoring process for the whole system. It usually has less overhead since it does not require to virtualize counters per process. This collection type can collect uncore counters and requires kernel configuration.
- **Driverless Perf per-process counting** provides event counting statistics over an interval for a single process or its children. Event counting can be done simultaneously from multiple monitoring processes. Since it requires performance counters virtualization per process, it can bring more overhead in comparison with system-wide collection. Typically, a system has this type of collection enabled by default.
- **Driverless Perf system-wide counting** provides event counting statistics performed by one monitoring process for the whole system over an interval. It usually has less overhead since it does not require to virtualize counters per process. This collection type can collect uncore counters and requires kernel configuration.

To configure system-wide driverless collection:

Set the `/proc/sys/kernel/perf_event_paranoid` value to 0 or less. Root privileges are required.

> **NOTE**
> For the kernel modules resolution, make sure you have enough permissions to read kernel symbols information from the `/proc/kallsyms` file.

To use only driverless collection, where possible:

Create a custom analysis and select the **Enable driverless collection** option.

From the command line, use the `-knob enable-driverless-collection=true` option. For example:

```
amplxe-cl -collect-with runsa -knob enable-stack-collection=true -knob stack-size=0 -knob enable-
user-tasks=true -knob event-config=CPU_CLK_UNHALTED.THREAD:sa=2000000,
INST_RETIRED.ANY:sa=2000000, -knob enable-driverless-collection=true
```

To disable the driverless collection for your analysis:

Create a custom analysis and set the **Stack size** option to 0 (unlimited) value.

From the command line, use the `-knob stack-size=0` option. For example:

```
amplxe-cl -collect-with runsa -knob enable-stack-collection=true -knob stack-size=0 -knob enable-
user-tasks=true -knob event-config=CPU_CLK_UNHALTED.THREAD:sa=2000000,INST_RETIRED.ANY:sa=2000000
```

This option disables the Perf driverless collection for stacks and enables the VTune Amplifier driver-based collection instead.

Analysis Limitations

Perf-based driverless collection is applicable to all hardware event-based sampling analysis types, such as Hotspots (hardware event-based sampling mode), Microarchitecture Exploration, and Custom event-based sampling analysis types on Linux and Android OS. If the uncore events support is available on the system, the VTune Amplifier also uses the Perf collection for Memory Access, HPC Performance Characterization, and Microarchitecture Exploration analysis types with the **Analyze memory bandwidth** option enabled.

The following additional limitations are also possible for the driverless collection:

- Since the driverless collection is based on the Linux Perf functionality, all Perf limitations fully apply to the VTune Amplifier sampling analysis as well. For example, your operating system limits on the maximum amount of files opened by a process as well as maximum memory mapped to a process address space still apply and may affect Perf-based profiling. For more information, see the *Tutorial: Troubleshooting and Tips* topic at https://perf.wiki.kernel.org/index.php/Main_Page.
- Local and remote **Launch Application**, **Attach to Process** and **Profile System** target types are supported but this support fully depends on the Linux Perf profiling credentials specified in the `/proc/sys/kernel/perf_event_paranoid` file and managed by the administrator of your system using root credentials. For more information, see *Perf Events and tool security* at https://www.kernel.org/doc/html/latest/admin-guide/perf-security.html and the *perf_event related configuration files* topic at http://man7.org/linux/man-pages/man2/perf_event_open.2.html. By default, only user processes profiling at both user and kernel spaces is permitted, so you need granting wider profiling credentials via the `perf_event_paranoid` file to employ the **Profile System** target type.
- Memory bandwidth analysis is not supported on Intel Atom® processors.
- Preemption and synchronization context switches may not be differentiated on kernels older than 4.17. To identify context switch types, make sure the VTune Amplifier sampling driver is loaded and the **Stack size** option is set to 0.

> **NOTE**
> Run the `<install-dir>/bin64/amplxe-self-checker.sh` script to explore the analysis type collection abilities of your system. The script output helps recognize limitations and provides advice on fixing them.

Platform Profiler Setup (Preview)

Use the Platform Profiler tool packaged with Intel VTune Amplifier to gain insights into overall system configuration, performance, and behavior. Before you begin collecting data, set up Platform Profiler and the target system on which you want to collect platform performance data.

> **NOTE**
> This is a **PREVIEW FEATURE**. A preview feature may or may not appear in a future production release. It is available for your use in the hopes that you will provide feedback on its usefulness and help determine its future. Data collected with a preview feature is not guaranteed to be backward compatible with future releases. Please send your feedback to parallel.studio.support@intel.com or to intelsystemstudio@intel.com.

Initial Configuration

1. Navigate to `<vtune_install_dir>/vpp/server`.

2. Run the following commands for the configuration script and follow the prompts using an account with root or sudo privileges.

```
./vpp-server config
./vpp-server start
```

3. Navigate to the Platform Profiler home page in your browser (Google Chrome* recommended): `http://localhost:6543`

Starting and Stopping Platform Profiler

To start Platform Profiler manually after initial installation or a system reboot, run:

```
<vtune_install_dir>/vpp/server/vpp-server start
```

To stop Platform Profiler, run:

```
<vtune_install_dir>/vpp/server/vpp-server stop
```

Collector Installation

The collectors used for the Platform Profiler are automatically installed with other VTune Amplifier collectors on a target system. No additional configuration is required.

Getting Started with Intel® VTune™ Amplifier

5

Information about using Intel© VTune™ Amplifier after installing the product is available from the Getting Started page, which is available from `<install-dir>/documentation/<language>/welcomepage/get_started.htm` or from the Intel Developer Zone: https://software.intel.com/en-us/get-started-with-vtune.

Intel® VTune™ Amplifier Installation FAQs

6

The following topics include answers to frequently asked questions about the Intel VTune Amplifier installation and configuration process.

How do I install the drivers required by Intel VTune Amplifier?

Intel VTune Amplifier automatically installs the required sampling drivers (SEP) when the product is installed, assuming the user who installed the product has the appropriate permissions (administrative/root/sudo access). Drivers are installed on a target system when VTune Amplifier connects to the target system during analysis configuration (administrative/root/sudo access required via passwordless SSH connection). If the drivers fail to install, they can be configured manually.

The Sampling Drivers section includes detailed information about installing the drivers for Linux*, Windows*, or Android* systems.

Error Message: Installing and Starting VTSS++ Driver...FAILED

Cause

When you install the drivers manually, the Intel® VTune™ Amplifier may display this message if memory in the system is insufficient.

Solution

Try rebooting the system.

Error Message: No Pre-built Driver Exists for This System

When executing the `build-driver` script on Linux*, you may see a warning message similar to the following if the kernel sources are not configured properly (they do not match the kernel that is running): `Warning: Current running kernel is version 2.4.18-e.31smp`. After successfully building the driver and running the `insmod-sep3` or `insmod-sep` command, the following message appears: `No pre-built driver exists for this system`.

Solution

To resolve this issue, execute the following commands to configure the kernel sources:

```
$ cd /usr/src/linux
$ make mrproper
$ cp /boot/config-'uname-r' .config
$ vi Makefile
```

Make sure that `EXTRAVERSION` matches the tail of the output of `uname -r`. The resulting `/user/src/linux/include/version.h` should have a `UTS_RELEASE` that matches the output of `uname -r`. Once that is true, run the following commands:

```
$ make oldconfig
```

```
$ make dep
```

After completing these steps, run the `build-driver` script to build the sampling driver against the kernel sources in `/usr/src/linux`

What features are available if I install Intel VTune Amplifier as a non-root or non-administrator user?

Many Intel VTune Amplifier features work when installed on a system as a user other than Root or Administrator. If VTune Amplifier is installed by a user with permissions other than root/administrative, then VTune Amplifier can collect some data. For example, VTune Amplifier may try to use the Perf* utility and run a hardware event-based sampling analysis in the driverless mode.

The following features require root/administrative access:

- Analysis on an Intel Atom® processor
- Installing the sampling driver (SEP)
- Running a performance analysis using an analysis type that uses event-based sampling (list available from https://software.intel.com/en-us/vtune-amplifier-help-hardware-event-based-sampling-collection)

How do I configure remote data collection using Intel VTune Amplifier?

Intel VTune Amplifier can collect performance data on the system on which it is installed (Windows* or Linux* only; macOS* data collection is not supported) or on a remote target system (Windows, Linux, or Android* targets). To analyze performance on a remote system, follow these general steps:

1. (Linux only) Configure password-less SSH access to the target system. Password-less SSH access is required for running a remote analysis because VTune Amplifier will not prompt for a password once analysis collection begins. For detailed steps, see Configuring SSH Access for Remote Collection.

   NOTE
   VTune Amplifier will not automatically install drivers on the remote system unless you connect via SSH as the root user. If you want to run an event-based collection, but do not want to set up password-less SSH access using the root user account, manually install the drivers on the target system. For detailed steps, see Sampling Drivers.

2. (Android only) Configure ADB access to the target system. For detailed steps, see Android System Setup .
3. Launch VTune Amplifier and open the **Configure Analysis** window.
4. Select the appropriate target system from the **WHERE** pane. For **Remote Linux (SSH)** targets, specify the username, hostname, and port in the **SSH Destination** field.

   VTune Amplifier connects to the remote target and installs appropriate collectors and drivers (if root access was provided). This can take a few seconds.
5. Specify a target type in the **WHAT** pane.
   - **Launch Application**: specify the location of the process on the target system.
   - **Attach to Process**: select a process from the list of available processes running on the target system.
   - **Launch Android Package**: select an application and corresponding package (`*.apk`) available on the target system from the list of available packages.
6. Select an analysis type from the **HOW** pane and click **Start** to run the analysis on the remote system.

For detailed steps for setting up remote collection on a specific target system, use the following topics:

- Linux System Setup for Remote Analysis
- Android System Setup

Can I install multiple versions of Intel VTune Amplifier on the same system?

You can install multiple major versions of VTune Amplifier on the same system, but you cannot have multiple installations of the same major version on the same system.

For example, you can have both VTune Amplifier 2018 Update 1 and VTune Amplifier 2019 on the same system. However, if you have VTune Amplifier 2018 Update 1 installed, installing VTune Amplifier 2018 Update 2 will uninstall Update 1 and replace it with the Update 2 content.

Problem: Platform Not Authorized

When attempting to use Intel® VTune™ Amplifier with a floating license, you may encounter errors such as:

```
(License server does not support this feature (-18,327))
```

```
(This platform not authorized by license (-89,337))
```

Cause

This error is caused by an outdated license server.

Solution

Login to the Intel® Registration Center to download the latest license server:

> **NOTE**
> The registered owner of the license must log in. Other registered users do not get access to the license server download.

1. Log in to the Intel® Registration Center.
2. In your supported products list, look for the entry for **Intel® Software License Manager** and click the version number next to it.

   You will be taken to the download page for the Intel® Software License Manager.
3. In the **Platform** box on the right, select the operating system for your license server, then download the appropriate package for your license server.
4. Install the new license server software on your license server.
5. Ensure that the new license server software is running on the license server.

This should resolve the error. If not, see the Get Help page for your support options.

Error Message: Cannot Collect GPU Hardware Metrics

Possible error messages:

- *Cannot collect GPU hardware metrics because `libmd.so` cannot be loaded. Make sure you have installed Metrics Discovery API from https://github.com/intel/metrics-discovery correctly.*
- *Cannot collect GPU hardware metrics because `libmd.so` was not found. Make sure you have installed Metrics Discovery Application Programming Interface from https://github.com/intel/metrics-discovery.*
- *Cannot collect GPU hardware metrics because your version of the Metrics Discovery API is obsolete.*

Cause

To collect GPU hardware metrics and GPU usage data on Linux, the VTune Amplifier uses the Intel® Metric Discovery API library distributed with the product. If it cannot access the library, the corresponding error message is provided.

Solution

Install the Intel Metric Discovery API library from the official repository at https://github.com/intel/metrics-discovery and make sure to meet the following requirements:

- To enable the VTune Amplifier to successfully load the library, it should be linked to `libstdc++` (version GLIBCXX_3.4.20 or older) or statically linked to `libstd++`. If `libmd.so` is dynamically linked to a newer version of `libstdc++`, make sure to have it loaded to the process before loading `libmd.so`. You can do this, for example, by re-defining the environment variable `LD_PRELOAD`:

  ```
  LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libstdc++.so.6 amplxe-cl -c gpu-hotspots.
  ```
- If you use `su` or `sudo` command to run the VTune Amplifier, you need to redefine `LD_PRELOAD` directly in the command, for example:

  ```
  sudo LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libstdc++.so.6 amplxe-cl -c gpu-hotspots
  ```

Uninstalling Intel® VTune™ Amplifier

7

Use the following steps to uninstall Intel® VTune™ Amplifier from your system:

NOTE
These steps should be completed by the same user who installed VTune Amplifier.

1. Open a command prompt and navigate to a folder outside of the VTune Amplifier installation directory.
2. Run one of the following commands:
   - To uninstall using the command prompt:

     `<install-dir>/vtune_amplifier_<version>.<number>.<package>/uninstall.sh`
   - To uninstall using the GUI:

     `<install-dir>/vtune_amplifier_<version>.<number>.<package>/uninstall_GUI.sh`
3. Follow the prompts in the command prompt or the user interface.

8 Additional Resources

The following online resources may be helpful during or after the installation of Intel® VTune™ Amplifier.

- Release Notes (https://software.intel.com/en-us/articles/intel-vtune-amplifier-release-notes): The VTune Amplifier release notes include helpful information specific to your release version, including system requirements and known issues and limitations.
- Intel VTune Amplifier User Guide (https://software.intel.com/en-us/vtune-amplifier-help): Learn about using VTune Amplifier with the online user guide.
- Troubleshooting Help (https://software.intel.com/en-us/vtune-amplifier-help-troubleshooting): This section of the VTune Amplifier online user guide includes solutions to common problems.
- Download Intel® Parallel Studio XE documentation | Download Intel® System Studio documentation (https://software.intel.com/en-us/articles/download-documentation-intel-parallel-studio-xe-current-previous and https://software.intel.com/en-us/articles/download-documentation-intel-system-studio-current-previous): Starting with the 2018 releases, most documentation is available on the web only. You can download a copy of all documentation for use without an internet connection.

Index

A

Android*
  application analysis *35*
  remote analysis *30*
  search directories *36*

B

build
  sampling drivers *52*

C

command line collectors *51*

D

debug information
  for Linux kernels *18*
disclaimer *4*
driverless sampling collection *57*
drivers
  build *16*
  for Android *30*
  for FreeBSD *29*
  Linux* *16*

E

embedded device *19*, *25*, *26*, *28*
Enter index keyword *38*, *48*, *59*, *62–64*, *67*
error messages
  failed to install vtss++ driver *62*
  no pre-built driver exists for this system *62*

F

FreeBSD*
  remote analysis *29*
FreeBSD* target *29*

G

guest system profiling *43*

I

install
  automated *9*
  cluster environment *10*
  command line *9*
  sampling drivers *52*
  user interface *8*

J

Java* code analysis
  enable on Android *30*

K

kernel modules resolution *18*
KVM guest OS profiling *43*

L

legal information *4*
load
  sampling drivers *52*

P

password-less mode *49*
Perf*-based collection *57*
prepare
  Android application for analysis *35*
  Android system for analysis *30*
  FreeBSD system for analysis *29*
  Linux* system for analysis *16*
problem
  platform not authorized *64*

R

remote analysis *16*, *29*, *30*, *49*
remote collectors *16*, *49*

S

sampling driver *57*
sampling drivers *52*, *56*
search directories
  for Android *36*
SSH access configuration *49*

T

target system
  Linux* *16*
  prepare for analysis *16*
  sampling drivers *52*
troubleshooting
  drivers *62*

V

virtual environment *43*
virtual machine performance analysis *36*

W

Wind River Linux *19*

Y

yocto *16*
Yocto Project *19*, *25*, *26*, *28*
Yocto* *52*