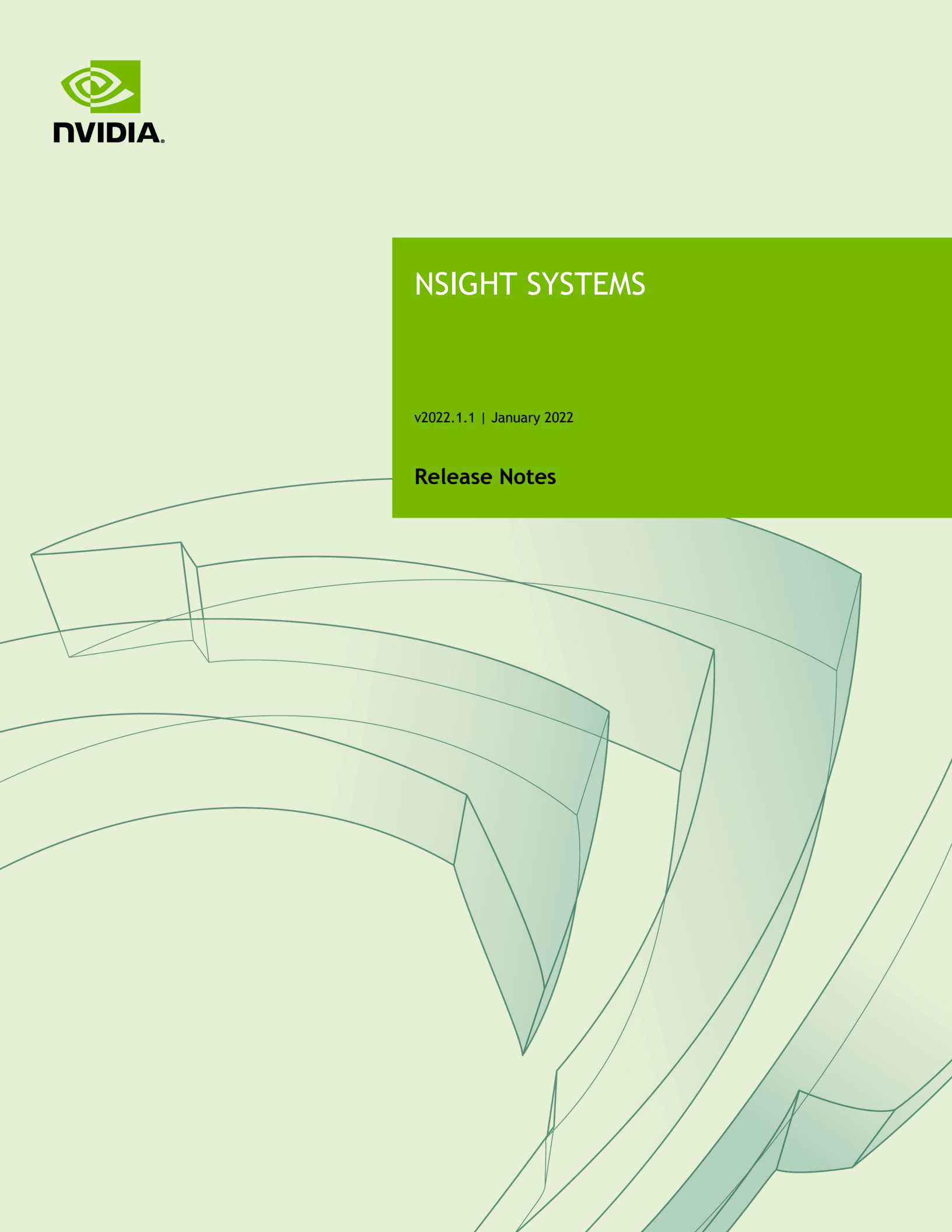

NSIGHT SYSTEMS

v2022.1.1 | January 2022

Release Notes

TABLE OF CONTENTS

Chapter 1. WHAT'S NEW

- System wide CPU sampling (CLI only).
- Export in Apache Arrow format file.
- Support for running nsys as sudo/root, but application being profiled as user.
- BETA: Added MPI communication parameter trace.

Chapter 2. KNOWN ISSUES

2.1. General Issues

- The current release of Nsight Systems CLI doesn't support naming a session with a name longer than 63 characters. Profiling an executable with a name exceeding 49 characters is also unsupported by the **`nsys profile`** command. Those limitations will be removed in a future version of the CLI.
- Nsight Systems 2020.4 introduces collection of thread scheduling information without full sampling. While this allows system information at a lower cost, it does add overhead. To turn off thread schedule information collection add --cpuctxsw=none to your command line or turn off in the GUI
- Profiling greater than 5 minutes is not officially supported at this time. Profiling high activity applications, on high performance machines, over a long analysis time can create large result files that may take a very long time to load, run out of memory, or lock up the system. If you have a complex application, we recommend starting with a short profiling session duration of no more than 5 minutes for your initial profile. If your application has a natural repeating pattern, often referred to as a frame, you may typically only need a few of these. This suggested limit will increase in future releases.
- Attaching or re-attaching to a process from the GUI is not supported with the x86_64 Linux or IBM Power target. Equivalent results can be obtained by using the interactive CLI to launch the process and then starting and stopping analysis at multiple points.
- To reduce overhead,

  Nsight Systems traces a subset of API calls likely to impact performance when tracing APIs rather than all possible calls. There is currently no way to change the subset being traced when using the CLI. See respective library portion of this documentation for a list of calls traced by default. The CLI limitation will be removed in a future version of the product.

- There is an upper bound on the default size used by the tool to record trace events during the collection. If you see the following diagnostic error, then Nsight Systems hit the upper limit.

```
Reached the size limit on recording trace events for this process.
       Try reducing the profiling duration or reduce the number of features
       traced.
```

- When profiling a framework or application that uses CUPTI, like some versions of TensorFlow(tm), Nsight Systems will not be able to trace CUDA usage due to limitations in CUPTI. These limitations will be corrected in a future version of CUPTI. Consider turning off the application's use of CUPTI if CUDA tracing is required.

  As an example, in the TensorFlow **`mnist_with_summaries.py`** tutorial, you will be able to use Nsight Systems to perform CUDA trace if you remove usage of **`RunOptions.FULL_TRACE`** from the code. For more information, see RunOptions documentation.
- Tracing an application that uses a memory allocator that is not thread-safe is not supported.
- Tracing OS Runtime libraries in an application that preloads glibc symbols is unsupported and can lead to undefined behavior.
- Nsight Systems cannot profile applications launched through a virtual window manager like GNU Screen.
- Using Nsight Systems MPI trace functionality with the Darshan runtime module can lead to segfaults. To resolve the issue, unload the module.

```
module unload darshan-runtime
```

- Profiling MPI Fortran APIs with MPI_Status as an argument, e.g. MPI_Recv, MPI_Test[all], MPI_Wait[all], can potentially cause memory corruption for MPICH versions 3.0.x. The reason is that the MPI_Status structure in MPICH 3.0.x has a different memory layout than in other MPICH versions (2.1.x and >=3.1.x have been tested) and the version (3.3.2) we used to compile the Nsight Systems MPI interception library.
- Using **`nsys export`** to export to an SQLite database will fail if the destination filesystem doesn't support file locking. The error message will mention:

```
std::exception::what: database is locked
```

- On some Linux systems when VNC is used, some widgets can be rendered incorrectly, or Nsight Systems can crash when opening Analysis Summary or Diagnostics Summary pages. In this case, try forcing a specific software renderer: **`GALLIUM_DRIVER=llvmpipe nsys-ui`**
- Due to a known bug in OpenMPI 4.0.1, target application may crash at the end of execution when being profiled by Nsight Systems. To avoid the issue, use a different OpenMPI version, or add **`--mca btl ^vader`** option to **`mpirun`** command line.

2.2. vGPU Issues

- When running Nsight Systems on vGPU you should always use the profiler grant. See Virtual GPU Software Documentation for details on enabling NVIDIA CUDA Toolkit profilers for NVIDIA vGPUs. Without the grant, unexpected migrations may crash a running session, report an error and abort. It may also silently produce

a corrupted report which may be unloadable or show inaccurate data with no warning.

- Starting with vGPU 13.0, device level metrics collection is exposed to end users even on vGPU. Device level metrics will give info about all the work being executed on the GPU. The work might be in the same VM or some other VM running on the same physical GPU.
- As of CUDA 11.4 and R470 TRD1 driver release, Nsight Systems is supported in a vGPU environment which requires a vGPU license. If the license is not obtained after 20 minutes, the tool will still work but the reported GPU performance metrics data will be inaccurate. This is because of a feature in vGPU environment which reduces performance but retains functionality as specified in Grid Licensing User Guide.

2.3. Docker Issues

- In a Docker, when a system's host utilizes a kernel older than v4.3, it is not possible for Nsight Systems to collect sampling data unless both the host and Docker are running a RHEL or CentOS operating system utilizing kernel version 3.10.1-693 or newer. A user override for this will be made available in a future version.
- When **`docker exec`** is called on a running container and stdout is kept open from a command invoked inside that shell, the exec shell hangs until the command exits. You can avoid this issue by running with **`docker exec --tty`**. See the bug reports at:
  - https://github.com/moby/moby/issues/33039
  - https://github.com/drud/ddev/issues/732

2.4. CUDA Trace Issues

- When using CUDA Toolkit 10.X, tracing of DtoD memory copy operations may result in a crash. To avoid this issue, update CUDA Toolkit to 11.X or the latest version.
- Nsight Systems will not trace kernels when a CDP (CUDA Dynamic Parallelism) kernel is found in a target application on Volta devices or later.
- On Tegra platforms, CUDA trace requires root privileges. Use the **Launch as root** checkbox in project settings to make the profiled application run as root.
- If the target application uses multiple streams from multiple threads, CUDA event buffers may not be released properly. In this case, you will see the following diagnostic error:

```
Couldn't allocate CUPTI bufer x times. Some CUPTI events may
       be missing.
```

  Please contact the Nsight Systems team.
- In this version of Nsight Systems, if you are starting and stopping profiling inside your application using the interactive CLI, the CUDA memory allocation graph generation is only guaranteed to be correct in the first profiling range. This limitation will be removed in a future version of the product.

- CUDA GPU trace collection requires a fraction of GPU memory. If your application utilizes all available GPU memory, CUDA trace might not work or can break your application. As an example cuDNN application can crash with **`CUDNN_STATUS_INTERNAL_ERROR`** error if GPU memory allocation fails.
- For older Linux kernels, prior to 4.4, when profiling very short-lived applications (~1 second) that exit in the middle of the profiling session, it is possible that Nsight Systems will not show the CUDA events on the timeline.
- When more than 64k serialized CUDA kernels and memory copies are executed in the application, you may encounter the following exception during profiling:

  ```
  InvalidArgumentException: "Wrong event order detected"
  ```

  Please upgrade to the CUDA 9.2 driver at minimum to avoid this problem. If you cannot upgrade, you can get a partial analysis, missing potentially a large fraction of CUDA events, by using the CLI.
- On Vibrante, when running a profiling session with multiple targets that are guest VMs in a CCC configuration behind a NAT, you may encounter an error with the following text during profiling:

  ```
  Failed to sync time on device.
  ```

  Please edit the group connection settings, select **Targets on the same SoC** checkbox there and try again.
- When using the 455 driver, as shipped with CUDA Tool Kit 11.1, and tracing CUDA with Nsight Systems you many encounter a crash when the application exits. To avoid this issue, end your profiling session before the application exits or update your driver.