

# RedHawk 9.2 OpenOnload Installation Instructions Release Notes

August 29, 2023



*This page intentionally left blank*

# 1. Introduction

This document provides instructions for installing OpenOnload version 8.1.1.17 for Onload compatible cards software onto an existing system installed with RedHawk Linux 9.2.x.

## 2. Requirements

- An x86\_64 host system running RedHawk 9.2.x
- A *RedHawk OpenOnload* version 9.2 DVD
- A supported Onload adapter card listed on page 43 of the *Onload User Guide*, available on the DVD.

## 3. Installation

### 3.1. Boot to RedHawk 9.2

Ensure that the system is booted with a RedHawk 9.2 kernel and invoke the following command to verify:

```
# uname -r
```

### 3.2. Build and install all OpenOnload software

Mount the *RedHawk OpenOnload* DVD and issue the following commands to install the various base distribution packages that are required by OpenOnload and then build and install all OpenOnload software:

```
# mkdir -p /tmp/mount
# mount /dev/cdrom /tmp/mount
# cd /tmp/mount
# ./install
```

#### NOTE

*This installation script can run for as long as 30 minutes as it compiles and installs kernel modules and several user-space packages and then updates the firmware in the Onload adapter card installed in the system. The many warnings produced during installation can be safely ignored.*

When the installation is complete, reboot the system with the same RedHawk kernel. After the reboot, proceed with the steps below.

### 3.3. Verify installation

Invoke the following command to verify that OpenOnload has been successfully installed:

```
# onload --version
```

Output like the following should be displayed:

```
Onload 8.1.1.17
Copyright (c) 2002-2023 Advanced Micro Devices, Inc.
Built: Aug 24 2023 12:17:07 (release)
Build profile header: <ci/internal/transport_config_opt_extra.h>
Kernel module: 8.1.1.17
```

### 3.4. Simple Onload configuration test

This section provides steps for verifying that connections between two directly connected compatible Onload adapter cards can be established successfully.

1. On both systems, modify the `/etc/hosts` file to define two addresses. In this simple example, one system is named *server* and the other system is named *client*. Add entries like the following: note that the text added must be identical on both systems:

```
10.1.1.1 server-oo
10.1.1.2 client-oo
```

2. On both systems, create a `/etc/sysconfig/network-scripts/ifcfg-<interface>` file, where `<interface>` is the name of the interface of the Onload adapter card.

```
NAME=<interface>
DEVICE=<interface>
BOOTPROTO=none
TYPE=Ethernet
IPADDR=10.1.1.1           ← Use 10.1.1.1 for server and 10.1.1.2 for client
NETMASK=255.255.255.0
NETWORK=10.1.1.0
BROADCAST=10.1.1.255
ONBOOT=no
```

3. Reboot both systems, ensuring that the same RedHawk kernels are booted.
4. Issue the following command on both systems to verify that the Onload network is configured correctly and that both hosts are active and connected:

```
# ifconfig <interface>
```

5. Finally, verify that you can ping and ssh from both systems to the other system:
  - ping client-oo from *server-oo*
  - ping server-oo from *client-oo*
  - ssh into client-oo from *server-oo*
  - ssh into server-oo from *client-oo*

This concludes the simple Onload configuration and connection verification.

Please contact Concurrent Real-Time technical support if you had any problems during this installation (<http://www.concurrent-rt.com/support> or 1-800-245-6453).

### 3.5. IP over OpenOnload Latency Testing

This section provides steps for verifying the connection between two directly connected Onload cards can achieve advertised latency.

Sfnettest is a set of benchmark tools and test utilities created by Solarflare for benchmark and performance testing of network servers and network adapters. On both systems, issue the following commands to build and install Sfnettest from source:

```
# git clone https://github.com/Xilinx-CNS/cns-sfnettest
# cd cns-sfnettest/src
# make
# cp sfnt-pingpong /usr/bin
```

First, set some configuration options that decrease latency for Onload acceleration technologies. On both machines:

```
# systemctl stop firewalld
# sysctl -w vm.nr_hugepages=1024
# ethtool -C <interface> rx-usecs 0 adaptive-rx off
```

It is also recommended to shield the CPU core and its NUMA node the test will run on. For example, on both systems CPU 4 and its NUMA node were shielded:

```
# shield -a C4
```

1. On the server run the following to start sfnt-pingpong on CPU 4:

```
# onload --profile=latency-best run -b4 sfnt-pingpong
```

2. On the client run the following to start the test on CPU 4:

```
# onload --profile=latency-best run -b4 sfnt-pingpong --affinity "4;4" \
tcp 10.1.1.1
```

Output like the following should appear on the client once the test establishes a connection to the server:

size	mean	min	median	max	%ile	stddev	iter
1	1829	1713	1825	7890	1998	50	813000
2	1835	1715	1829	14982	2002	53	811000
4	1832	1708	1828	6394	2000	49	813000
8	1828	1716	1825	5071	1998	48	814000
16	1831	1715	1827	6659	2000	50	813000
32	1844	1720	1839	4935	2018	51	807000
64	1859	1741	1854	11697	2019	51	801000
128	1891	1776	1888	7314	2028	48	787000
256	1984	1834	1983	5061	2105	42	751000
512	2111	1936	2105	5963	2255	52	706000
1024	2234	2079	2228	6244	2377	51	667000
2048	3514	3228	3500	13018	3860	109	425000
4096	4213	3938	4203	8774	4489	98	355000
8192	5801	5390	5794	9160	6142	116	258000
16384	8347	7757	8339	13166	8873	180	180000
32768	13649	12036	13640	17128	14914	547	110000
65536	23276	20703	23094	29703	25917	1080	65000

The output identifies mean, minimum, median and maximum (nanosecond) ½ RTT latency for increasing packet sizes, including the 99% percentile and standard deviation for these results.