The science behind the report:

**Improving energy efficiency in the data center: Endure higher temperatures with confidence with Dell PowerEdge HS5620 servers**

This document describes what we tested, how we tested, and what we found. To learn how these facts translate into real-world benefits, read the report *Improving energy efficiency in the data center: Endure higher temperatures with confidence with Dell PowerEdge HS5620 servers*.

We concluded our hands-on testing on April 9, 2024. During testing, we determined the appropriate hardware and software configurations and applied updates as they became available. The results in this report reflect configurations that we finalized on March 11, 2024 or earlier. Unavoidably, these configurations may not represent the latest versions available when this report appears.

**Charts**

Figure 1: SSD and processor temperatures in the Supermicro® SYS-621C-TN12 during an additional fan failure scenario, where the server ran a floating-point workload with fan 3 disabled. The workload began at 0:15 and ended at 2:15. SSD 1 ran the OS, while SSD 2 was idle. Source: Principled Technologies.
Figure 2: Processor core speeds of the Dell™ PowerEdge™ HS5620 and the Supermicro SYS-621C-TN12 during the first fan failure scenario. The floating-point workload began at 0:15 and ended at 2:15. Source: Principled Technologies.

Figure 3: Processor core speeds of the Dell PowerEdge HS5620 and the Supermicro SYS-621C-TN12 during the malfunctioning HVAC scenario. The floating-point workload began at 0:15 and ended at 2:15. Source: Principled Technologies.
# System configuration information

Table 1: Detailed information on the systems we tested.

<table>
<thead>
<tr>
<th>System configuration information</th>
<th>Dell PowerEdge HS5620</th>
<th>Supermicro SYS-621C-TN12R</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIOS name and version</td>
<td>Dell 2.1.3</td>
<td>Supermicro 2.1</td>
</tr>
<tr>
<td>Non-default BIOS settings</td>
<td>Performance Per Watt (OS)</td>
<td>N/A</td>
</tr>
<tr>
<td>Operating system name and version/build number</td>
<td>Ubuntu 22.04.3</td>
<td>Ubuntu 22.04.3</td>
</tr>
<tr>
<td>Date of last OS updates/patches applied</td>
<td>3/11/24</td>
<td>1/28/2024</td>
</tr>
<tr>
<td>Power management policy</td>
<td>Performance Per Watt (OS)</td>
<td>Balanced Performance</td>
</tr>
</tbody>
</table>

### Processor

<table>
<thead>
<tr>
<th>Number of processors</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vendor and model</td>
<td>Intel® Xeon® Gold 6444Y</td>
<td>Intel Xeon Gold 6444Y</td>
</tr>
<tr>
<td>Core count (per processor)</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>Core frequency (GHz)</td>
<td>3.60 (4.0 turbo)</td>
<td>3.60 (4.0 turbo)</td>
</tr>
<tr>
<td>Stepping</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>

### Memory module(s)

| Total memory in system (GB) | 1,024 | 1,024 |
| Number of memory modules   | 16 | 16 |
| Vendor and model            | Hynix® HMCG94AEBRA109N | SK Hynix HMCG94MEBRA123N |
| Size (GB)                   | 64 | 64 |
| Type                        | DDR5 DIMM | DDR5 |
| Speed (MHz)                 | 4,800 | 4,800 |
| Speed running in the server (MHz) | 4,800 | 4,800 |

### Storage controller (front storage)

<table>
<thead>
<tr>
<th>Vendor and model</th>
<th>Dell HBA355i Adp</th>
<th>Supermicro MegaRAID AOC-S3916L-H16R-32DD-P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache size (GB)</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Firmware version</td>
<td>24.15.14.00</td>
<td>5.240.02-3768</td>
</tr>
<tr>
<td>Driver version</td>
<td>N/A</td>
<td>52.24.0-4766</td>
</tr>
</tbody>
</table>

### Storage controller (NVMe® M.2)

<table>
<thead>
<tr>
<th>Vendor and model</th>
<th>Dell BOSS-N1 Monolithic</th>
<th>N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache size (GB)</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>Firmware version</td>
<td>2.1.13.2025</td>
<td>N/A</td>
</tr>
<tr>
<td>Local storage (OS)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Number of drives

<p>| Number of drives | 2 | 2 |
| Drive vendor and model | Dell NVMe PE8010 RI M.2 960GB | Micron® 7450 MTFDKBA960TFR |</p>
<table>
<thead>
<tr>
<th>System configuration information</th>
<th>Dell PowerEdge HS5620</th>
<th>Supermicro SYS-621C-TN12R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drive size (GB)</td>
<td>960</td>
<td>960</td>
</tr>
<tr>
<td>Drive information (speed, interface, type)</td>
<td>8GT/s M.2 SSD</td>
<td>PCIe® M.2 NVMe</td>
</tr>
<tr>
<td>Local storage (data)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of drives</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Drive vendor and model</td>
<td>HGST HUH721212AL5200</td>
<td>WDC WUH721814ALE6L4</td>
</tr>
<tr>
<td>Drive size (GB)</td>
<td>120,000</td>
<td>1,400</td>
</tr>
<tr>
<td>Drive information (speed, interface, type)</td>
<td>12 Gbps SAS 3.5&quot; HDD</td>
<td>6Gb SATA 3.5&quot; HDD</td>
</tr>
<tr>
<td>Network adapter A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor and model</td>
<td>4x Intel 25G 2P E810-XXV</td>
<td>3x Intel E810-XXVAM2 (AOC-S25GC-i2S)</td>
</tr>
<tr>
<td>Number and type of ports</td>
<td>2x 25Gb</td>
<td>2x 25Gb</td>
</tr>
<tr>
<td>Driver version</td>
<td>22.5.7</td>
<td>4.20 (0x800177B4)</td>
</tr>
<tr>
<td>Network adapter B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor and model</td>
<td>1x Broadcom® NetXtreme Gigabit Ethernet (BCM5720)</td>
<td>1x Intel E810-XXVAM2 (AOC-A25G-i2SM)</td>
</tr>
<tr>
<td>Number and type of ports</td>
<td>2x 1Gb</td>
<td>2x 25Gb</td>
</tr>
<tr>
<td>Driver version</td>
<td>22.71.3</td>
<td>4.30 (0x800177B4)</td>
</tr>
<tr>
<td>Cooling fans</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number, vendor, model</td>
<td>1x Dell HPR Gold</td>
<td>3x Supermicro Middle Fan FAN-0206L4</td>
</tr>
<tr>
<td></td>
<td>5x Dell HPR Silver</td>
<td></td>
</tr>
<tr>
<td>Power supplies</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor and model</td>
<td>Dell 05222NA00</td>
<td>Supermicro PWS-1K23A-1R</td>
</tr>
<tr>
<td>Number of power supplies</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Wattage of each (W)</td>
<td>1,800</td>
<td>1,200</td>
</tr>
</tbody>
</table>
How we tested

To create an environment for which we could control and measure the temperature, we built a custom enclosure around a fully loaded 42U server rack. We tested both the Dell and Supermicro systems in the same position in the rack, and we collected internal server temperatures during three scenario types. We ran the stress-ng benchmark on the servers in the rack in four waves, each spaced apart by 1 minute and 10 seconds. The Dell and Supermicro systems we tested launched the workload in the fourth wave, starting 3 minutes and 30 seconds after the first servers began the workload. Below, we outline the steps we took to set up and run the tests.

Installing and configuring Ubuntu 22.04.3

1. Boot from Ubuntu 22.04.3 media.
2. Select Try or Install Ubuntu Server.
3. At the language menu, leave the defaults, and select Done.
4. Select Update to the new installer.
5. At the keyboard configuration, leave the defaults, and click Done.
6. At the network connections menu, leave the defaults, and select Done.
7. At the configure proxy screen, leave the defaults, and select Done.
8. At the configure Ubuntu archive mirror screen, wait for the test to pass, and select Done.
9. At the guided storage configuration screen, leave the defaults, and select Done.
10. At the storage configuration summary screen, leave the defaults, and select Done.
11. At the storage configuration summary screen, leave the defaults, and select Done.
12. To confirm destructive action, select Continue.
13. At the profile setup screen, under Your name and Username, enter ptuser. Under Your servers name, and enter a name, and confirm a password.
14. Select Done.
15. At the upgrade to Ubuntu Pro screen, leave the defaults, and select Continue.
16. At the SSH setup screen, select Install OpenSSH server, and select Done.
17. At the featured server snaps screen, leave the defaults, and select Done.
18. When the installation is complete, select Reboot now.
19. Log into Ubuntu using the credentials you created above.
20. Process updates:

   ```
   sudo apt update
   sudo apt upgrade
   ```

21. Install CIFS Utilities and map the PT share:

   ```
   sudo apt install cifs-utils
   sudo mkdir /mnt/pt-data01
   sudo mount -t cifs //10.41.1.21/pt /mnt/pt-data01/ -o "rw,user=<useraccount>,pass=<password>"
   ```

22. Configure networking:

   ```
   sudo cp /etc/netplan/*.yaml /etc/netplan/00-installer-config.yaml.bak
   sudo nano /etc/netplan/*.yaml
   ```

   ```
   sudo apt install cifs-utils
   sudo mkdir /mnt/pt-data01
   sudo mount -t cifs //10.41.1.21/pt /mnt/pt-data01/ -o "rw,user=<useraccount>,pass=<password>"
   ```

23. Identify desired network adapter and make the following adjustments:

   ```
   addresses:
   - <IP_Address>/<CIDR>
   routes:
   - to: default
     via: <Default_Gateway>
   nameservers:
     search: [<NameServer1>, <NameServer2>]
     addresses: [<DNS_IP1>, <DNS_IP2>, <DNS_IP3>]
   ```
24. Test and apply the changed file:

```bash
sudo netplan try
sudo netplan apply
```

25. Set the hostname:

```bash
sudo hostnamectl set-hostname <NewHostname>
```

26. Restart the host:

```bash
sudo shutdown -r now
```

**Deploying password-less sudo**

**Deploying on the client side**

1. Edit the sudoers file:

```bash
sudo visudo /etc/sudoers
```

2. Add the following at the very end of the file:

```bash
ptuser ALL=(ALL:ALL) NOPASSWD: ALL
```

**Deploying on the controller side**

1. Generate the SSH keypair:

```bash
ssh-keygen -t rsa -b 4096 -N "" -f "$HOME/.ssh/id_rsa.pub"
```

2. Copy the SSH public key to each remote server:

```bash
ssh-copy-id ptuser@<remote_server_IP>
```

**Implementing TIG-P stack for data collection**

**Configuring Docker**

1. Log into the logging machine as ptuser.

2. Prepare for Docker installation:

```bash
sudo apt update
sudo apt install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc
```

3. Add the repository to Apt sources and install:

```bash
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.
```
Configuring Huntabyte TIG stack

1. On the logging machine, clone the tig-stack repository:
   ```
git clone https://github.com/huntabyte/tig-stack.git
   ```

2. Edit the .env file for your deployment:
   ```
sudo nano tig-stack/.env
   ```

3. For FluxDB, fill in the username, password, org, bucket, and retention period as follows:
   ```
DOCKER_INFLUXDB_INIT_USERNAME: admin
DOCKER_INFLUXDB_INIT_PASSWORD: <PasswordHere>
DOCKER_INFLUXDB_INIT_ORG: PT
DOCKER_INFLUXDB_INIT_BUCKET: <BucketName>
DOCKER_INFLUXDB_INIT_RETENTION: 52w
   ```

4. Generate a random 32-character hex string with the following command, and enter the result for the admin token in the .env file:
   ```
openssl rand -hex 32
   ```

5. Save, and exit.

6. Edit telegraf.conf:
   ```
sudo nano tig-stack/telegraf/telegraf.conf
   ```

7. Set the following values:
   ```
services:
influxdb:
  image: influxdb
telegraf:
  image: gibletron/telegraf-ipmitool
grafana:
  image: grafana/grafana-oss
  links:
    - prometheus
   ```

8. Save, and exit.

9. Start Docker compose (headless / detached):
   ```
sudo docker-compose up -d
   ```

10. On each server you wish to monitor, run the following command:
    ```
sudo apt install telegraf
    ```

11. To open the InfluxDB management interface, browse to the IP address of the InfluxDB host with port 8086.
12. Create API token(s) as needed, making sure to record them before closing the window.
13. Under Load Data, click API Tokens, and click Generate API Token.
14. On each server you wish to monitor, edit /etc/telegraf/telegraf.conf with the following:

   ```
   [[outputs.influxdb_v2]]
   url = ["<influxDB_IP>:8086"]
   token = "<API_token>"
   organization = "PT"
   bucket = "<bucket_name>"
   ```

15. On each system under test, add the following:

   ```
   [[inputs.intel_powerstat]]
   cpu_metrics = ["cpu_frequency"]
   ```

16. Save, and exit.
17. Restart Telegraf:

   ```
   sudo systemctl restart telegraf
   ```

### Configuring Prometheus

1. Add the following to /home/ptuser/tig-stack/docker-compose.yml:

   ```yaml
   prometheus:
   image: prom/prometheus:latest
   volumes:
   - ${{PROM_CFG_PATH}}:/etc/prometheus/prometheus.yml
   - prom-storage:/prometheus
   ports:
   - 9090:9090
   volumes:
   prom-storage:
   ```

2. Save, and exit.
3. Edit the .env file, and add the following:

   ```bash
   PROM_CFG_PATH=./prometheus/prometheus.yml
   ```

4. Save, and exit.
5. On each server you wish to monitor, run the following command:

   ```
   sudo apt install dbus prometheus-node-exporter prometheus-node-exporter-collectors -y
   ```

6. On each system under test, run the following command:

   ```
   sudo apt install prometheus -y
   ```

7. To create the monitoring job in Prometheus, add the following to the /home/ptuser/tig-stack/prometheus/prometheus.yml:

   ```yaml
   - job_name: "<custom_name>
     static_configs:
     - targets: ["<target_IP:9090>"]
   ```

8. Add additional jobs and/or targets by creating additional entries similar to step 7. You can add other targets for the same job as another target line.
9. Save, and exit.
Testing with stress-ng

In each test scenario, we followed these steps to run the stress-ng floating-point workload.

1. On each server, run the following command:

```bash
sudo apt install stress-ng linux-tools-generic -y
```

2. On each server under test, run the following commands:

```bash
sudo modprobe rapl
sudo modprobe intel_rapl_common
sudo modprobe intel_rapl_msr
sudo modprobe msr
sudo modprobe intel-uncore-frequency
sudo setcap cap_sys_rawio,cap_dac_read_search,cap_sys_admin+ep /usr/bin/telegraf
sudo chmod -R a+r /sys/devices/virtual/powercap/intel-rapl/
```

3. On each server under test, navigate to https://github.com/andikleen/pmu-tools/blob/master/event_download.py, download the raw file, and execute it:

```bash
sudo chmod +x event_download.py
./event_download.py
```

4. On the controller, install PSSH:

```bash
sudo apt install pssh -y
```

5. On the controller, create files to utilize during the execution of stress-ng:

```bash
sudo touch ~/.pssh_hosts_file
sudo touch ~/.pssh_hosts_file_wave1
sudo touch ~/.pssh_hosts_file_wave2
sudo touch ~/.pssh_hosts_file_wave3
sudo touch ~/.pssh_hosts_file_wave4
```

6. Edit the file ~/.pssh_hosts_file, and enter all server IP addresses with one on each line.
7. Edit the files ~/.pssh_hosts_file_wave1 through ~/.pssh_hosts_file_wave4, and enter appropriately one fourth of the IP addresses in each file.
8. Test that all servers are online and responding to remote commands:

```bash
sudo pssh -i -h ~/.pssh_hosts_file uptime
```

9. On the controller, create a log folder for the stress-ng test:

```bash
sudo mkdir /var/log/stress-ng
sudo chmod 777 /var/log/stress-ng
```

10. Execute a test with the following commands, editing "wave1" with the appropriate wave number.

```bash
pssh -i -h ~/.pssh_hosts_file_wave1 sudo stress-ng --cpu 4 --matrix 0 --cpu-method matrixprod --mq 4 --hdd 6 --tz --metrics --perf --times --aggressive -t 2h --log-file /var/log/stress-ng/$(date +"%Y%m%d_%H%M%S").log
```
This project was commissioned by Dell Technologies.