1. ABSTRACT

Publishing scientific results without the detailed execution environments describing how the results were collected makes it difficult or even impossible for the reader to reproduce the work. However, the configurations of the execution environments are too complex to be described easily by authors. To solve this problem, we propose a framework facilitating the conduct of reproducible research by tracking, creating, and preserving the comprehensive execution environments with Umbrella. The framework includes a lightweight, persistent and deployable execution environment specification, an execution engine which creates the specified execution environments, and an archiver which archives an execution environment into persistent storage services like Amazon S3 and Open Science Framework (OSF). The execution engine utilizes sandbox techniques like virtual machines (VMs), Linux containers and user-space tracers, to create an execution environment, and allows common dependencies like base OS images to be shared by sandboxes for different applications.

We evaluate our framework by utilizing it to reproduce three scientific applications from epidemiology, scene rendering, and high energy physics. We evaluate the time and space overheads of reproducing these applications using different sandbox techniques – Parrot, Docker and the Amazon EC2. Our results show that these applications can be reproduced using different sandbox techniques successfully and efficiently, even through the overhead and performance slightly vary.

2. Tracking Execution Environment: Umbrella Spec

```
{...
  "description": "A ray-tracing application which creates video frames.",
  "hardware": {
    "arch": "x86_64",
    "os": "Ubuntu 18.04",
    "memory": "32GB",
    "disk": "500GB"
  }
  ...
}
```

3. Creating Execution Environment: Umbrella Engine

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Kernel</th>
<th>OS</th>
<th>Sandbox Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Utilize the current OS directly (Fig. 3)</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>OS-level virtualization (Docker, Parrot) (Fig. 3)</td>
</tr>
</tbody>
</table>

```
2. Umbrella reads and parses povray.umbrella
4. Umbrella sends the Umbrella job to the EC2 instance via scp
6. Umbrella returns the experiment results
8. Umbrella returns the experiment results

Fig. 3. Workflow of Umbrella Execution Engine (local)

4. Preserving Execution Environment: Umbrella Archiver

```
1. Alice composes and tests povray_local.umbrella using local dependencies
3. Alice publishes the OSF URL of povray_osf.umbrella
5. Bob downloads povray.osf.umbrella to his machine, and reproduces Alice's work
7. povray.osf.umbrella is archived by the Umbrella Archiver
```

```
5.2 Overheads of Creating Execution Environments

```
<table>
<thead>
<tr>
<th>Application</th>
<th>OpenMalaria</th>
<th>Povray</th>
<th>CMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS Deps</td>
<td>Software Deps</td>
<td>Data Deps</td>
<td></td>
</tr>
<tr>
<td>OpenMalaria</td>
<td>CentOS 6.8</td>
<td>povray</td>
<td>povray</td>
</tr>
<tr>
<td></td>
<td>6 (605MB/1.8GB)</td>
<td>6.5 (605MB/1.8GB)</td>
<td>6.5 (605MB/1.8GB)</td>
</tr>
<tr>
<td></td>
<td>openMalaria (2.9GB/135MB)</td>
<td>povray (1.5GB/2.9GB)</td>
<td>povray (1.5GB/2.9GB)</td>
</tr>
<tr>
<td></td>
<td>rpm packages (203MB)</td>
<td>povray (1.5GB/2.9GB)</td>
<td>povray (1.5GB/2.9GB)</td>
</tr>
<tr>
<td></td>
<td>awp1.repo (&lt;1KB)</td>
<td>povray (1.5GB/2.9GB)</td>
<td>povray (1.5GB/2.9GB)</td>
</tr>
</tbody>
</table>

5.5 Last step to Enhance Reproducibility – DOI

```
Information on this webpage:
DOI info: 
Link to the Umbrella specification file
Links to the OSF deops
Links to the software deops
Links to the data deops
Links to the Umbrella installation docs
Link to the Umbrella user manual
Link to the experiment result
```

The University of Notre Dame Center for Research Computing scientists and engineers provided critical technical assistance throughout this research effort.