Introduction to Reprozip
Why Reprozip?

● Hard for reviewers to reproduce results because of:
  different OS
  missing libraries
  different library versions
  issues in installing required dependencies (dependency hell)

● Reprozip solves the problem by:
  capturing all the necessary component
  providing different unpacking methods and interfaces that
  avoids the need to install all the required dependencies
  making it possible to run the experiment under different inputs
What is Reprozip?

- ReproZip is a tool for creating reproducible experiments from command-line executions.
- Captures required provenance of existing experiments.
  - Uses captured provenance to:
    - Create self-contained reproducible results.
    - Include binaries, data and dependencies.
    - Derive a workflow specification for the experiment.
- Readers/reviewers can extract the package and execute the workflow to reproduce and explore the experiment.
Packing Experiments

Experiment → Execute → Reprozip

Capture Provenance

- Command line arguments
- Working Directories
- Input files
- Output files
## Software Requirements

<table>
<thead>
<tr>
<th>Component</th>
<th>Linux</th>
<th>Mac OS</th>
<th>Windows</th>
</tr>
</thead>
<tbody>
<tr>
<td>reprozip</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>reprounzip</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Python 2.7.3 or greater is required to run ReproZip

<table>
<thead>
<tr>
<th>Components/plugins</th>
<th>Required Software Packages</th>
</tr>
</thead>
<tbody>
<tr>
<td>reprozip</td>
<td>SQLite, a working C compiler</td>
</tr>
<tr>
<td>reprounzip</td>
<td>None</td>
</tr>
</tbody>
</table>
Obtaining the Software on Linux/Ubuntu machine

$ pip install reprozip
$ pip install reprounzip

sudo apt-get install python-pip
sudo apt-get install python-dev
sudo apt-get install sqlite3
sudo apt-get install libssqlite3-dev
Using reprozip

- Trace an Experiment
- Create a Package
- Email the reprozip package to another environment
- Unzip the package using reprounzip
Trace an experiment

- $ reprozip trace <command-line>
  <command-line> is the command line used to execute the experiment.
  Traces the operating system calls used by the experiment
  Identifies all the necessary information for its future re-execution
  Automatically identifies the main output files generated by the experiment

- $reprozip trace --dont-identify-packages <command-line>
  Automatically identifies the distribution packages, to avoid this use above command:
Editing the Configuration File

- `.reprozip-trace/config.yml` created by the tracer, and drives the packing step.
- change the command line parameters by editing `<command-line-arguments>`
- add or remove environment variables by editing `<environment-variables>`.
- software dependencies identified are in the packages section:
  - packages:
Create a Package

- $ reprozip pack <package-name>
  - <package-name> is the name you want to give to the package.
- Files will be copied from your environment and into the package
- This package can be sent to others for reproducing the same results
Using reprounzip

1) Inspecting a package:
   ● $ reprounzip info <package>
     <package> corresponds to the experiment package
   ● The output of this command has three sections.
     ----- Pack information -----  
     ----- Metadata -----  
     ----- Unpackers -----  

2) Showing Input and Output Files

$ reprounzip showfiles package.rpz

Input files:
  program_config
  ipython_config
  input_data

Output files:
  rendered_image
  logfile
3) Unpacking an Experiment in Linux

$ reprounzip directory setup <package> <path>
➢ where <path> is the directory where the experiment will be unpacked.
➢ unpack the entire experiment (including library dependencies) in a single directory
➢ reproduce the experiment directly from the directory
➢ $ reprounzip directory run <path>
      will execute the entire experiment inside the experiment directory.
➢ Users may also change the command line of the experiment by using the argument *cmdline*:
➢ $ reprounzip directory run <path> --cmdline <new-command-line>
An example

- `reprozip trace java -jar repro-pack.jar`
- `reprozip pack repro-pack`
- `reprounzip showfiles repro-pack.rpz`
- `reprounzip info repro-pack.rpz`
- `reprounzip directory setup repro-pack.rpz ~/try_reprounzip`
- `reprounzip directory run ~/try_reprounzip`
- `java -jar pqsrepro-pack.jar`
Limitations

- Works only on Linux
- Package may not run if:
  - Underlying software is incompatible with target environment
  - Executables that use hard-coded paths
Conclusion

- Reprozip simplifies the creation of reproducible experiments
- Captures provenance and identifies components needed to reproduce results
- Users can customize the process
- Integrated with scientific workflows
- Reviewers can explore the experiment get review provenance for free
References

- Documentation
- Research Summary
  http://cds.nyu.edu/projects/reprozip/
- Video Lecture
  https://www.usenix.org/conference/tapp13/technical-sessions/presentation/chirigati