

# Using Containers to Validate Research on Confidential Data at Scale

Lars Vilhuber

May 2024

# Introduction

# Concerns

Concerns about confidentiality in statistical products have increased in the past several years:

- New disclosure avoidance techniques in the Decennial Census garnered much attention (an understatement...),
- also concerns about formal disclosure avoidance techniques for public-use microdata files (PUMFs) (see [Census Bureau delayed implementation of such methods for the American Community Survey](#))

# Validation and Verification

On the other hand,

- long-running pilot projects with (non-formal) synthetic microdata products (SynLBD, SIPP Synthetic Beta) came to an end in 2022 (End of life for the Cornell Synthetic Data Server September 30, 2022)
- not sure how active OPM Verification server was (Barrientos et al (2018))

# Scaling up

These pilot projects were not set up to scale, and yet they demonstrated that there is a need for such a process.

# Background

# Anecdotal evidence with SDS

From conversations/informal surveys:

- researchers were happy with the ability to access data without having to request a full-blown project in an FSRDC
- somewhat frustrated by the process (slowness)

# Reproducibility and SDS

SDS validation required typically substantial human debugging

- Reason: problems with the reproducibility of code in the social sciences despite similarity of environment.



## Intermediate causes

- no strong pre-testing of reproducibility, often intense use of interactive programming practices
- divergence in environments over time
- divergence of data schemas over time

### **Failure to maintain strong links**

## Some broader evidence

In a sample of over **8,000 replication packages** associated with high-profile economics articles, **only 30% had some sort of master script.**

# Other systems

Statistical agencies and research institutes have explored various ways to scale up access to confidential data, without full (remote) access to confidential data.

- Statistics Canada: Real Time Remote Access (RTRA) process,
- Norway: Microdata.no system,
- Germany/IAB: JoSuA system

# Access restrictions

Most such processes have limitations, including in their utility for general purpose analysis

Most still have some **strong access limitations**

- RTRA: organizational application process
- microdata.no: Institutional MOU (and only Norwegian residents)
- IAB: proposal process

# Analysis restrictions

Many systems strongly **limit the type of analysis** that is feasible by

- RTRA: restricting the software keywords that can be used, subset of SAS allowing to “calculate frequencies, means, percentiles, percent distribution, proportions, ratios and shares.
- microdata.no: by creating a structured new statistical language (albeit with increasingly sophisticated capabilities)

# Comparison

The comparison researchers and analysts make is (for right or wrong) to the **unfettered use of public-use data...**

# The quest

# Direct access is expensive

Remote-access or local secure access in the form of physical or virtual secure data enclaves is still the dominant - **but expensive** - way to access confidential data.

*The dominant method of access thus forces researchers to choose between lower quality data in an environment that corresponds to their preferred computing method, and higher quality confidential data in environments that are expensive for researchers, data providers, or both.*



**Possible solution**

# Containerized validation

- **Containers,**
  - hosted on public cloud platform or run on researcher laptop
  - provide access to synthetic or “plausible” data, and coding resources
  - mechanism to ensure authors can validate reproducibility of analysis
- Then submitted to the **confidential computing environment.**
  - analysis modified to use confidential data
  - enables a wide spectrum of plug-in disclosure avoidance measures as well
    - similar in spirit: IAB JoSuA system, but without hosting costs

# Containers in the wild

One of the first mentions of containers for scientific research was Boettiger (2015).

- **CodeOcean** is a commercial service facilitating that process by making the resources available through a web browser
- **Wholetale** and **MyBinder** are other (academically oriented) services that provide similar functionality<sup>1</sup>
- Many universities HPC clusters provide some support (Apptainer more popular than Docker)

# Containers in Social Sciences are still a challenge!

In a sample of over **8,000 replication packages** associated with high-profile economics articles, **only 11 had a Dockerfile** (the key build script for containers).

(That's  $n=11$ , not 11% - in fact, it's **0.13%** of replication packages.)

# What's new

The use of containers in this way is novel as a systematic way to provide **scalable, potentially high-throughput validation**, and differs in usage from previous methods, such as the Cornell Synthetic Data Server.

*I believe that it is promising as a modern way of implementing validation when data are confidential.*

# User perspective

# Use provided container with pre-provisioned data

Possibilities:

- use directly (*safer*)
- use as input to build own container (addition of components)

# Critically

Pre-provisioned data does not need to be “**analytically valid**” - need only be “**plausible**”!

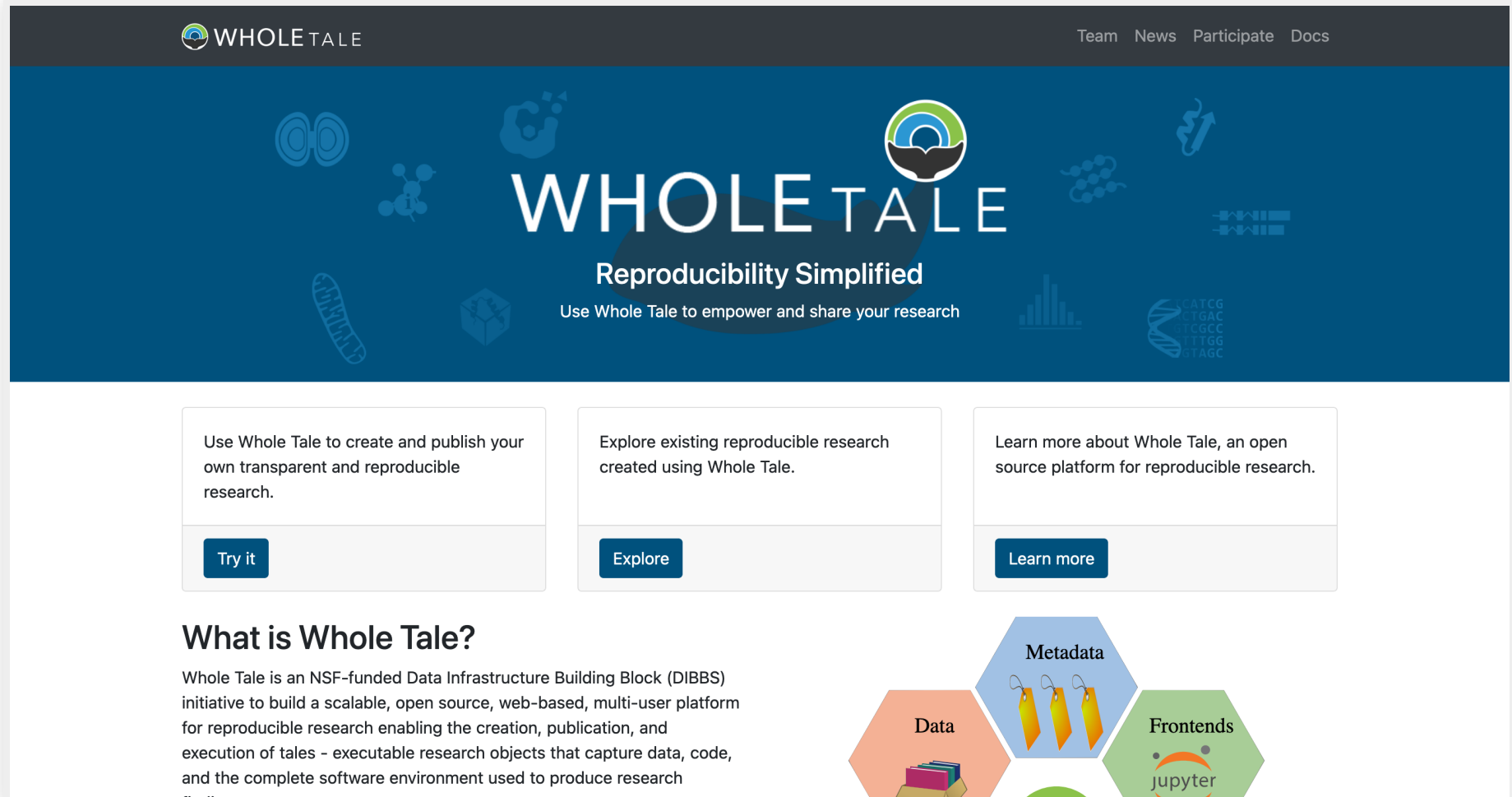


# Develop where feasible

Containers are generalized technology

- can be run on provisioned university computing infrastructure (most HPC systems can run containers)
- can run on desktops as needed (free container software available for all major operating systems for non-commercial use)
- can run on generic cloud infrastructure (AWS, Google Cloud, Azure)
- can run on custom cloud infrastructure specialized in running containers (Nuvolos, Codeocean, etc.)
- can be prepared by research institutions for use on their custom infrastructure (e.g., NSF-funded Whole Tale project)

# Whole Tale



The image shows the Whole Tale website. At the top is a dark navigation bar with the Whole Tale logo on the left and links for Team, News, Participate, and Docs on the right. Below this is a large blue banner with the text 'WHOLE TALE' in large white letters, 'Reproducibility Simplified' in smaller white letters, and 'Use Whole Tale to empower and share your research' in even smaller white letters. The banner is decorated with various scientific icons like a microscope, a DNA helix, a bar chart, and a molecular structure. Below the banner are three white boxes, each with a description and a button. The first box says 'Use Whole Tale to create and publish your own transparent and reproducible research.' with a 'Try it' button. The second box says 'Explore existing reproducible research created using Whole Tale.' with an 'Explore' button. The third box says 'Learn more about Whole Tale, an open source platform for reproducible research.' with a 'Learn more' button. Below these boxes is a section titled 'What is Whole Tale?' with a paragraph of text. To the right of the text is a diagram with three hexagons labeled 'Data', 'Metadata', and 'Frontends'. The 'Data' hexagon is orange and contains an icon of a stack of books. The 'Metadata' hexagon is blue and contains an icon of three yellow tags. The 'Frontends' hexagon is green and contains the Jupyter logo.

WHOLE TALE

Team News Participate Docs

**WHOLE TALE**

Reproducibility Simplified

Use Whole Tale to empower and share your research

Use Whole Tale to create and publish your own transparent and reproducible research.

[Try it](#)

Explore existing reproducible research created using Whole Tale.

[Explore](#)

Learn more about Whole Tale, an open source platform for reproducible research.

[Learn more](#)

## What is Whole Tale?

Whole Tale is an NSF-funded Data Infrastructure Building Block (DIBBS) initiative to build a scalable, open source, web-based, multi-user platform for reproducible research enabling the creation, publication, and execution of tales - executable research objects that capture data, code, and the complete software environment used to produce research findings.

**Data**

**Metadata**

**Frontends**

jupyter

# Cost to user

Cost: \$0 to low \$

# Run a container from the command line

```
## read the file run_docker.sh  
tail(readLines("run_docker.sh"),n=1)
```

# Run a container from Codeocean

The screenshot displays the Codeocean interface for a new capsule. The top bar shows the capsule is 'Private' and titled 'Untitled Capsule May 15, 2024 18:47'. Below this is a menu bar with 'Capsule', 'File', and 'Help'. The left sidebar contains icons for adding files, app builder, database, and a terminal. The main area is divided into three sections: 'Files', 'App Builder', and 'Tabs'.

**Files Section:** This section shows the file structure of the capsule. It includes a 'Core Files' section with 'metadata' (62 B) and 'environment' (219 B). The 'code' folder (92 B) is expanded, showing 'main.do' (92 B). The 'data' folder (0 B) is also expanded, with a 'Manage' button next to it. The 'Results' section is empty, and the 'Other Files' section is also empty.

**App Builder Section:** This section shows the 'main.do' file selected. The code is as follows:

```
1 /* Start of my code */
2
3 do "01_prepare_data.do"
4 do "02_analyis.do"
5 do "03_create_figures.do"
```

## Reproducible Run

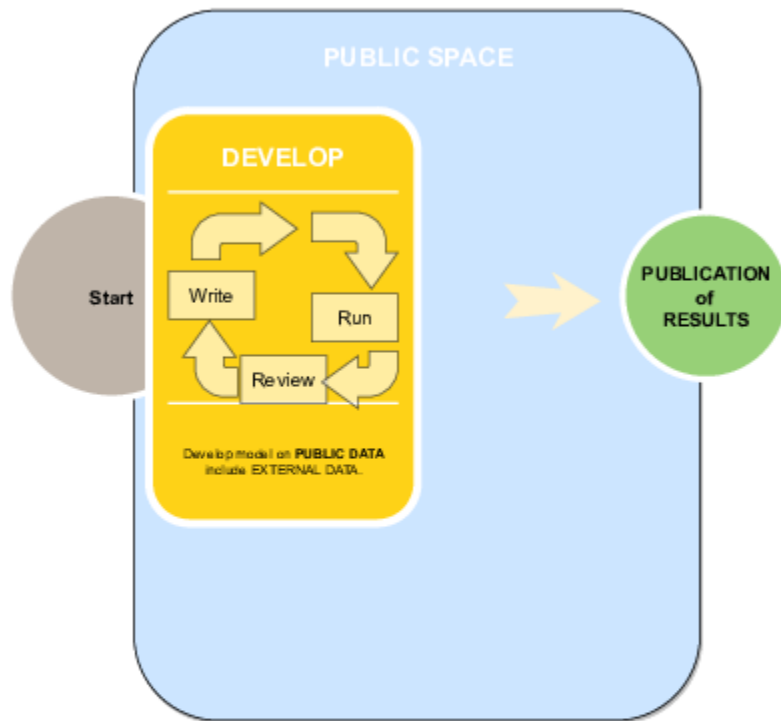
or launch a cloud workstation



Timeline

# Develop at will

- Arbitrary Stata, R, Python, etc. code





# **Provider perspective: Secure build**

# First impressions



# Internal build

- Prepare an internal container, compliant with IT security standards
  - secure configuration of container running system (base system)
  - add layer of common software (Stata, R, Python, various combinations) for **analysis system**
  - test suite (scripted) for updates

# Ability to leverage existing experience

- Can leverage existing container recipes for well-known software packages (rocker for R containers, datascience containers)
- Can leverage existing containers and harden the OS (if necessary)
- Already has process in place to securely vet imported libraries and packages - can be **reused**

# Public build

- Public “recipe” is the same as for internal
  - possibly up to secure base container - close enough is good enough
  - built by StatAgency itself

# Example: Build internal analysis system

```
FROM registry.internal.statagency.gov/os/ubuntu-24.04-secured  
  
# Install Stata from internal sources (simple tar file), no license  
...  
# Install R from internal sources  
...  
USER rstudio
```

# Example: Build public analysis system

```
FROM ubuntu-24.04

# Install Stata from internal sources (simple tar file), no license
...
# Install R from internal sources
...
USER rstudio
```

# Optional elements

While not strictly necessary, containers might contain

- development environments (Stata GUI, Jupyter notebooks, Rstudio)
- standard set of libraries (Stata ado files, R libraries, Python packages)




# Public posting

Prepared containers and recipes can be posted on public registries:

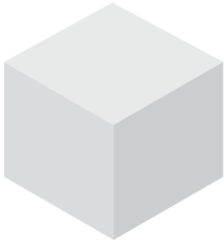
- post container on public registry ([Docker Hub](#), [Google Container Registry](#), etc.)
- post recipe on public repository ([GitHub](#), [GitLab](#), etc.)

# Posted on Docker Hub

 dockerhub

Search Docker Hub

[Explore](#) / dataeditors/stata17



## dataeditors/stata17 ☆9

By [dataeditors](#) • Updated 2 months ago

Docker image for Stata, to be used in automation and reproducibility.

[IMAGE](#)

[Overview](#) [Tags](#)

### Docker image basic Stata image

---

#### Purpose

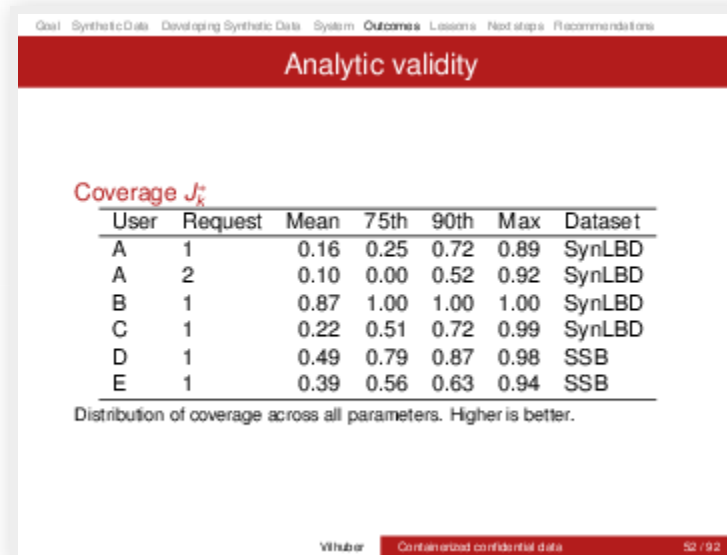
---

This Docker image is meant to isolate and stabilize that environment, and should be portable across multiple operating system, as long as [Docker](#) is available.

To learn more about the use of containers for research reproducibility, see [Carpentries' docker-](#)

# Also required: data

But if validation and verification are a key part of it, then data quality can be lower (**plausible, not analytically valid**)



The slide is titled 'Analytic validity' and features a table of coverage metrics. The table has columns for User, Request, Mean, 75th, 90th, Max, and Dataset. The data is as follows:

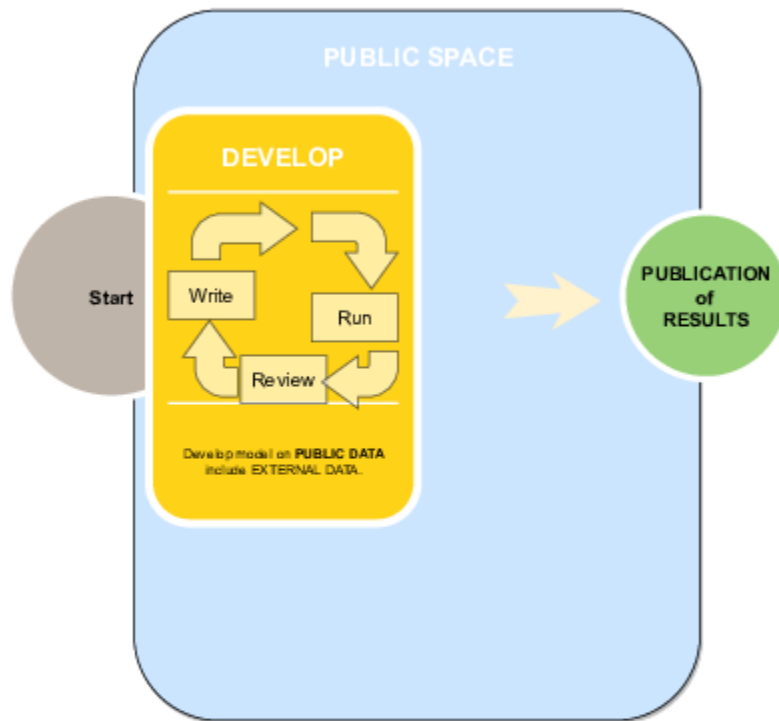
User	Request	Mean	75th	90th	Max	Dataset
A	1	0.16	0.25	0.72	0.89	SynLBD
A	2	0.10	0.00	0.52	0.92	SynLBD
B	1	0.87	1.00	1.00	1.00	SynLBD
C	1	0.22	0.51	0.72	0.99	SynLBD
D	1	0.49	0.79	0.87	0.98	SSB
E	1	0.39	0.56	0.63	0.94	SSB

Distribution of coverage across all parameters. Higher is better.

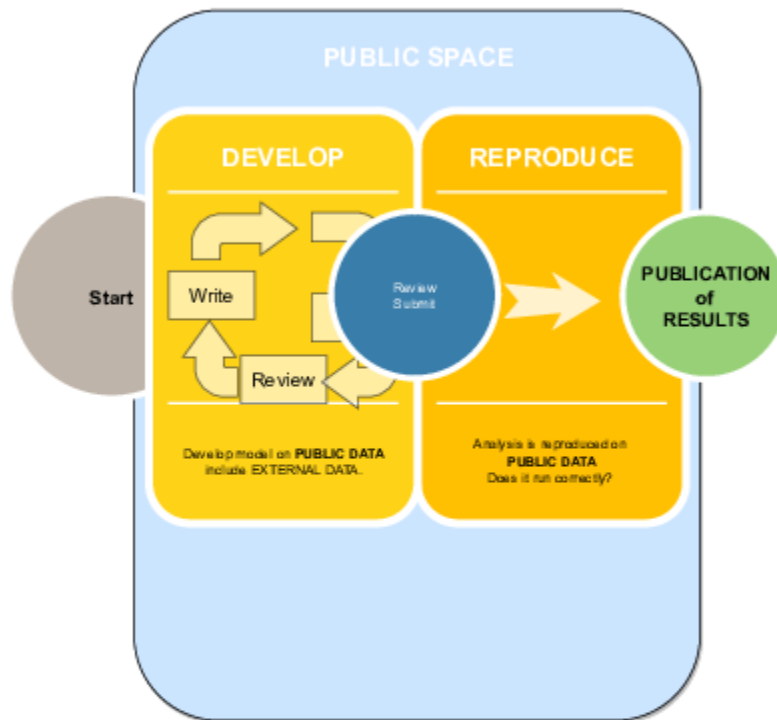
Whuber Contained confidential data 52 / 92

# Validation

# User Develops



# User tests



# User Submits for Validation

- Submit container *recipe* (Dockerfile) and code for validation to *StatAgency.gov*

```
./Dockerfile  
./code/01_prepare_data.R  
./code/02_run_analysis.R  
./code/03_create_figures.R
```

# **Important security aspect**

**No binary code is transmitted**

Any external data may need to be vetted.



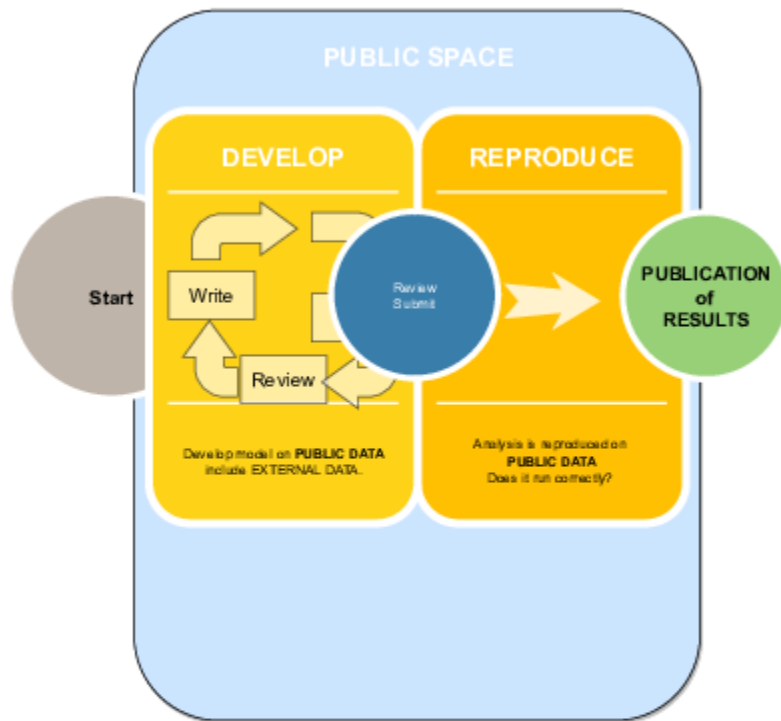
# *StatAgency* Upon receipt of submission

(**Automated**) system receives and processes

```
./Dockerfile  
./code/01_prepare_data.R  
./code/02_run_analysis.R  
./code/03_create_figures.R
```

# *StatAgency* validates reproducibility

Just to check that user actually did test...



## Provider validates reproducibility

If rejected, **automated system** returns to user without further ado.

If accepted, proceed to validation step

# Provider rebuilds container using secure base image

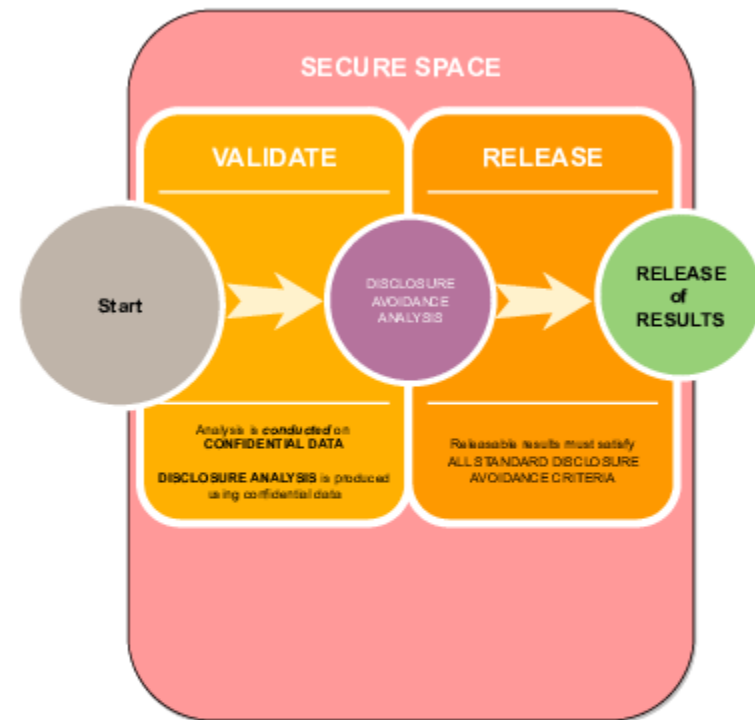
- Input is only the Dockerfile recipe
- Security scanning of (plaintext) scripts and of resulting image
- Build can occur in a sandboxed environment

# Necessary restrictions

While useful in the public space, when running internally and for pre-vetting,

- containers would be restricted in terms of internet access
- containers may be built against only known safe sources of packages (e.g. internal mirrors)

# Once image is built

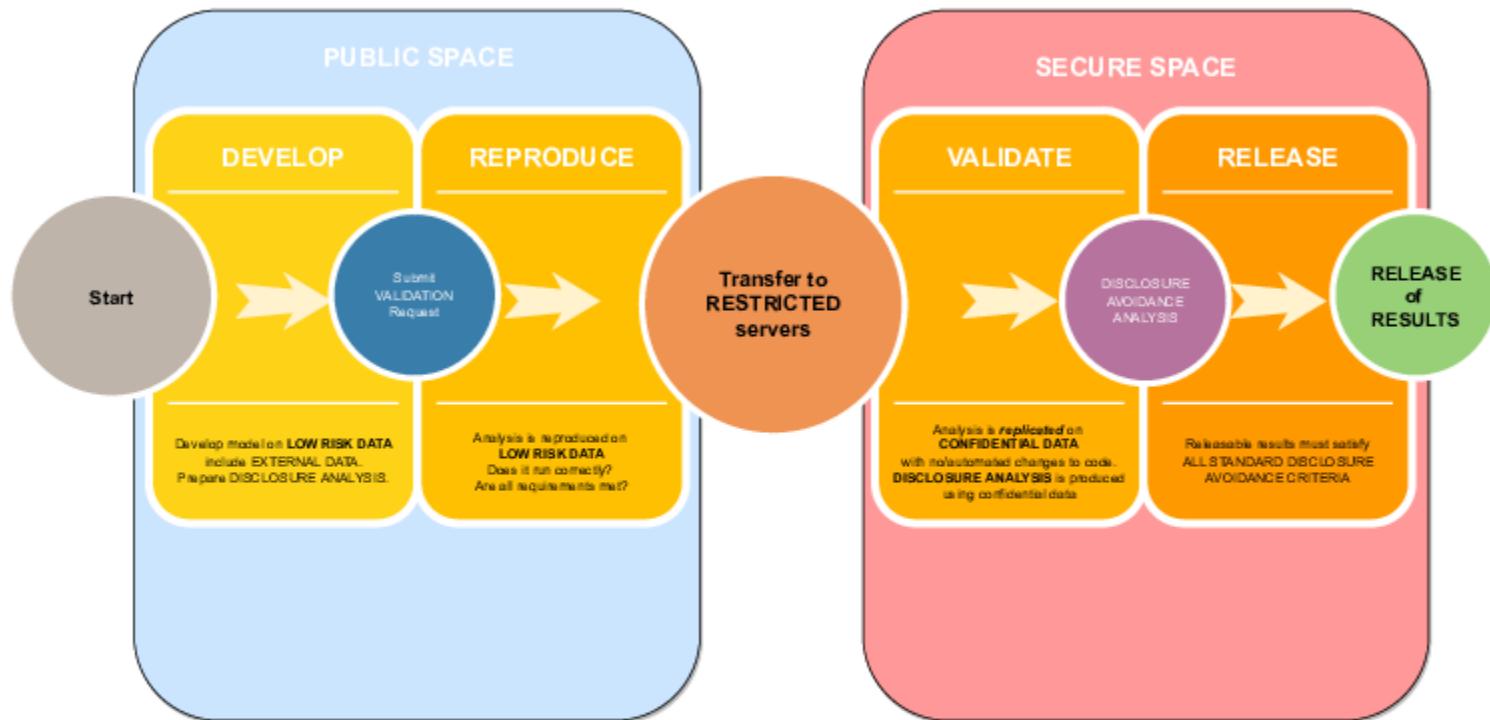


# Validate against confidential data

- Same image is used for confidential data
- Only difference: swap out public (test) data for confidential data
- Processing may involve more complex processing, for instance **bootstrapping errors** or **obtaining multiple estimates** across various partially protected datasets
- Disclosure avoidance may involve transparently **modifying certain functions**, or **post-processing of results**



# Return results to user



# Challenges

# Automation or streamlining of disclosure avoidance

Scalability of a system **hinges critically on streamlined output vetting.**

However, the challenge of creating automated and reliable disclosure avoidance procedures *is not unique* to the validation process described here.

# Security of containers

In general, bad idea to blindly run untrusted containers. However, this is a **solved problem** in the industry, facilitated by the (expected) sparsity of the build process.

# User acceptance

As a reminder, most social scientists are **not familiar with containers.**

- **Mitigation:**
  - Off-the-shelf solutions (Codeocean, Whole Tale)
  - IT support at universities and research institutions

# Advantages

# Existing technology

- Containers are well-known technology, including in other sciences
- Used by online services (Codeocean, but also Overleaf, etc.)

# Scalability

- Easy to scale to large number of users
- Easy to scale to technologies that allow for sophisticated but computing intensive disclosure avoidance



# Cheap

- for users
  - most of the core enabling technology is free to use
  - support by university IT is generally available
- for providers (*StatAgency*)
  - no need to provision scaled infrastructure for users
  - can leverage existing on-site software stacks (e.g., assuming that anything used internally is already security-vetted)

## Additional benefit

- **StatAgency** can accumulate a library of confirmed reproducible containers and models, and can test out new data, disclosure methods, etc. at scale against prior scientific findings

### **Consider the new disclosure avoidance method for ACS 2035**

- Can be tested against every submitted model that used prior ACS, as long as database schema is the same.

**Thank you**

# References

Boettiger, Carl. 2015. "An Introduction to Docker for Reproducible Research." *ACM SIGOPS Operating Systems Review* 49 (1): 71–79.  
<https://doi.org/10.1145/2723872.2723882>.

- 
1. An earlier version of this presentation mentioned Gigantum. As is not unusual in this space, Gigantum no longer functions as a company.↩
  2. Image credit Christopher Scholz, under CC-BY-SA 2.0↩