Towards True Reproducibility of Findings in Cybersecurity Research
Disclaimer
Design with the spirit of LASER in mind

How to capture
the essence of a keynote
without making it feel like
a speech or lecture?

(c) NY state Department of Education
Disclaimer
Design with the spirit of LASER in mind
Disclaimer
Design with the spirit of LASER in mind

key·note
/ˈkēˌnōt/
noun
a prevailing tone or central theme, typically one set or introduced at the start of a conference.
Disclaimer
Design with the spirit of LASER in mind

Work in progress  Speculation  Incomplete Ideas  Foster Discussion

Selfishly…
Outline

Oral History of Artifact Evaluation (student perspective)

Evaluators produce replicates

Language design for reproducibility
Formal languages unlock great power
Backstory

• September 2022: different workshop…

• Full disclosure… completely forgot!

• 2013/14 — lab mates participated in one of the earliest AECs for SIGPLAN

• 2014 — submitted artifact (OOPSLA 2014)

• 2014 — began AEC review (POPL 2015)

HOWTO for AEC Submitters

(last updated May 2022)

Dan Barowy (dbarowy@cs.umass.edu) - now at Williams College
Charlie Curtisinger (charlie@cs.umass.edu) - now at Grinnell College
Emma Touch (etouch@cs.umass.edu) - now University of Vermont
John Vill (jvill@cs.umass.edu) - now at Stripe

of the PLASMA group (http://plasma.cs.umass.edu) at University of Massachusetts Amherst

with encouragement and support from Emery Berger (emery@cs.umass.edu)

After serving on several Artifact Evaluation Committees and winning two Distinguished Artifact Awards, we put together this HOWTO document to help you submit an artifact that will pass the AEC process with flying colors.

How to Build a Good Software Artifact

1. Provide documentation with your artifact. We recommend that you prepare a Getting Started Guide. It should explain:
   a. how to download your artifact
   b. how to install your artifact
   c. how to run your artifact
   d. how to compare your artifact’s outputs to outputs described in your paper

2. Explicitly enumerate your claims in both your paper and in your artifact’s documentation.

3. Provide a VM if possible, and when appropriate. VMs aid reproducibility because they help control for nuisance factors that are not central to an author’s claims, significantly facilitating the review process. Nonetheless, reviewers may need to accept performance tradeoffs for VMs (e.g., because of the absence of special hardware). These tradeoffs are acceptable as long as authors explain to reviewers how and why they should adjust their expectations.

4. Provide step-by-step instructions, but make it easy for reviewers to supply their own inputs to your artifact. When reviewers can “play” with your artifact, it gives them confidence that your ideas were implemented robustly.

Source Code

1. If you are not bound by a nondisclosure agreement, make every effort to supply reviewers with source code. Good reviewers may read and modify your source code to learn the true capabilities of your artifact.

2. Document your code. You should sufficiently explain what is going on so that people who want to build on your work can do so.

3. If you discuss a new algorithm or unique implementation approach in your paper, have a reference to its implementation in the source code.
Have you ever served on an AEC committee or submitted an artifact to an AEC?

- Both served and submitted: 21%
- Only served: 7%
- Only submitted: 21%
- Neither: 43%
- What's Artifact Evaluation?: 7%

Powered by Poll Everywhere
The idea of submitting to artifact evaluation makes me feel...
The idea of serving on an artifact evaluation committee makes me feel..

stressed  overwhelmed  deadweight
upside  responsible  arrogant
clueless  excited  unprepared
big  time  need
overburdened  supportive
The rise and fall of expectations
Student perspective

Process early on

1. Read abstract, note expectations set by abstract
2. Read paper, revise expectations, in light of the paper
3. Write out expected software components and datasets*
4. Sketch a plan for something novel to do with the software
5. Early days: no separate guide
Student perspective

Reality

• Retrospective: assumed goal was reusability
  • Then: one badge. Now: Five

• Arguments in favor (at the time)
  • Promote best practices
  • Disincentivize “runs on my machine”
  • Temper reader’s expectations (inflated abstracts)
Only ever submitted once...
Why do we cite papers in the first place?

To please Reviewer #2.
Why do we cite papers in the first place?

• Findings
  • Don’t want to have to start from scratch

• Contributions
  • New Software
  • New Datasets
  • New Methods
  • New Research Areas

Is AE all and no ?

Oral History of Artifact Evaluation (student perspective)
Software will be cited if it works*  
...regardless of AE results

- Incentive: Public artifact
  - Don’t need artifact eval
  - Do we even want users?
    - Parable of SurveyMan

- Incentive: Good citizenship
  - Stand on the shoulder of giants!
  - Have you ever used someone else’s artifact? (Not repo)

Oral History of Artifact Evaluation (student perspective)
Not able to convince collaborators to submit
What about student evaluators?
Student perspective
(Students: feel free to share your thoughts)

• I liked serving on AECs
  • I learned new technologies
  • Reading others’ code makes your code better
  • Scalable training in methods
• Other incentives:
  • Be on a PC (now students officially on PCs)
  • Early on: part of something important
• Problem: evaluation is a lot of work
How to find more appealing carrots?

What do student stakeholders want out of the process?
Artifact Evaluators as contributors

Proposition 1
Submit...

something else

Proposition 2
Outline

Oral History of Artifact Evaluation (student perspective)

Evaluators produce replicates

Language design for reproducibility
Where's the carrot?

Answer: In empirical evaluation.
Focus efforts on replication
Eval as improvement to the science
The value of replicates...

DISCLAIMER: Work in Progress

Producing Wrong Data Without Doing Anything Obviously Wrong!

Todd Mytkowicz Amer Diwan
Department of Computer Science University of Colorado Boulder, CO, USA
{mytkowicz,diwan}@colorado.edu
Mathias Hauswirth
Faculty of Informatics University of Lugano Lugano, CH
Mathias.Hauswirth@unil.ch
Peter F. Sweeney
IBM Research Hawthorne, NY, USA
pfswen@us.ibm.com

Abstract
This paper presents a surprising result: changing a seemingly
innocuous aspect of an experimental setup can cause a sys-
tem research paper to draw wrong conclusions from an exper-
iment. What appears to be an innocuous aspect in the experi-
mental setup may in fact introduce a significant bias in an evalua-
tion. This phenomenon is called measurement bias in the
natural and social sciences.
Our results demonstrate that measurement bias is signif-
ificant and commonplace in computer system evaluation. By
significance, we mean that measurement bias can lead to a per-
fomance analysis that either overstates an effect or even
yields an incorrect conclusion. By commonplace we mean
that measurement bias occurs in all architectures that we
studied (Precise 4, Core 5, and m5 G3CPU), both compilers
that we tried (gcc and Intel’s C compiler), and most of the
SPEC CPU2006 C programs. Thus, we cannot ignore mea-
surement bias. Nevertheless, in a literature survey of 133 re-
cent papers from ASPLOS, PACT, PLDI, and CGO, we de-
termined that none of the papers with experimental results
adequately consider measurement bias.
Inspired by similar problems and their solutions in other
sciences, we describe and demonstrate two methods, one
for detecting (causal analysis) and one for avoiding (setup
cannot be done) measurement bias.
Categories and Subject Descriptors C.4 Computer Systems
Organization [C.4.4 Performance of Systems]: Design studies

General Terms Experimentation, Measurement, Performance

Keywords Measurement; Bias; Performance

Permission to make digital or hard copies of all or part of this work for personal or
classroom use is granted without fee provided that copies are not made or distributed
for profit or commercial advantage and that copies bear this notice and the full cita-
tion on the first page. Copyrights for components of this work owned by others than
ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or repub-
lish, to post on servers or to redistribute to lists, requires prior specific permission and/or
a fee.
ACM’s copyright policy permits individual reprints to be distributed free of charge.
Copyright © 2009 ACM 978-1-60558-056-5/09/06 $10.00.

1. Introduction
Systems researchers often use experiments to drive their
work: they use experiments to identify bottlenecks and then
again to determine if their optimizations for addressing the
bottlenecks are effective. If the experiments are biased, a
researcher may draw an incorrect conclusion: she may end
up wasting time on something that is not really a problem
and may conclude that her optimization is beneficial even
when it is not.
We show that experimental setups are often biased. For
example, consider a researcher who wants to determine if
optimization O is beneficial for system S. If she measures
S and S + O in an experimental setup that favors S + O,
she may overstate the effect of O or even conclude that O
is beneficial even when it is not. This phenomenon is called
measurement bias in the natural and social sciences. This
paper shows that measurement bias is commonplace and
significant: it can easily lead to a performance analysis that
yields incorrect conclusions.
To understand the impact of measurement bias, we inves-
tigate, as an example, whether or not Q3 optimizations are
beneficial to program performance when the experimental
setups differ. Specifically, we consider experimental setups
that differ along two dimensions: (i) UNIX environment size
(i.e., total number of bytes required to store the environ-
ment variables) because it affects the alignment of stack
distributed data; and (ii) link order (the order of -o flag that
we give to the linker) because it affects code and data layout.
There are numerous ways of affecting memory layout; we
decided to pick two to make the points in this paper but we
have found similar phenomena with the others that we have tried.
We show that changing the experimental setup often leads
to contradictory conclusions about the speedup of O3. By
“speedup of Q3” we mean run time with optimization level
Q3 divided by run time with optimization level O3. To in-
crease the generality of our results, we present data from two
microprocessors, Pentium 4 and Core 2, and one simulator,
m5 G3CPU [2]. To ensure that our results are not limited to
gcc, we show that the same phenomena also appear when we
use Intel’s C compiler.
Focus efforts on replication
Eval as improvement to the science

The value of replicates...
Specify as a causal graphical model

Two values; assign equal weight
Belief

Evaluators produce replicates
Focus efforts on replication
Eval as improvement to the science

The value of replicates…

Specify as a causal graphical model

Problems:

- Belief
- Performance

Over what???

Producing Wrong Data Without Doing Anything Obviously Wrong!

Todd Mytkowicz  Amer Diwan  Matthias Hauswirth  Peter F. Sweeney
Department of Computer Science  Faculty of Informatics  IBM Research
University of Colorado  University of Lugano  Hawthorne, NY, USA
Boulder, CO, USA  Lugano, CH
(mytkowit,diwan)@colorado.edu  Matthias.Hauswirth@unil.ch

Evaluators produce replicates
Focus efforts on replication
Eval as improvement to the science
The value of replicates...
Specify as a causal graphical model

- Belief
- Performance
- Population of all possible programs on my machine
- Population of all possible programs on all suitable machines

Producing Wrong Data Without Doing Anything Obviously Wrong!

Todd Mytkowicz  Amer Diwan
Department of Computer Science
University of Colorado Boulder, CO, USA
(mytkowicz.diwani)@colorado.edu

Matthias Hauswirth
Faculty of Informatics
University of Lugano Lugano, CH
Matthias.Hauswirth@unil.ch

Peter F. Sweeney
IBM Research
Hawthorne, NY, USA
pfSweeney.ibm.com
Focus efforts on replication
Eval as improvement to the science

The value of replicates…

Specify as a causal graphical model
Focus efforts on replication
Eval as improvement to the science

The value of replicates...

Specify as a causal graphical model
Focus efforts on replication
Eval as improvement to the science
The value of replicates…
Specify as a causal graphical model

<table>
<thead>
<tr>
<th>Programs x Machines</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programs</td>
<td>Y</td>
</tr>
</tbody>
</table>

Population of all possible programs on my machine
Population of all possible programs on all suitable machines

Belief

Evaluators produce replicates

Todd Mytkowicz  Amer Diwan
Department of Computer Science
University of Colorado
Boulder, CO, USA
(mytkowicz;diwan)@colorado.edu

Matthias Hauswirth
Faculty of Informatics
University of Lugano
Lugano, CH
Matthias.Hauswirth@unil.ch

Peter F. Sweeney
IBM Research
Hawthorne, NY, USA
pfsweeney.ibm.com
Focus efforts on replication
Eval as improvement to the science
The value of replicates...
Specify as a causal graphical model

Producers Wrong Data Without Doing Anything Obviously Wrong!

Todd Mytkowicz  Amer Diwan
Department of Computer Science
University of Colorado
Boulder, CO, USA
{mytkowicz, diwan}@colorado.edu

Matthias Hauswirth
Faculty of Informatics
University of Lugano
Lugano, CH
Matthias.Hauswirth@unil.ch

Peter F. Sweeney
IBM Research
Hawthorne, NY, USA
pfsweeney.ibm.com
Focus efforts on replication
Eval as improvement to the science

The value of replicates...

Specify as a causal graphical model

Sample is fixed! (Benchmark)
Focus efforts on replication
Eval as improvement to the science
The value of replicates…
Specify as a causal graphical model
Focus efforts on replication
Eval as improvement to the science
The value of replicates…
Specify as a causal graphical model

Sample is fixed! (Benchmark)

Producing Wrong Data Without Doing Anything Obviously Wrong!

Evaluator produce replicates

Todd Mytkowicz  Amer Diwan
Department of Computer Science
University of Colorado
Boulder, CO, USA
(mytkowicz@colorado.edu)

Matthias Hauswirth
Faculty of Informatics
University of Lugano
Lugano, CH
Matthias.Hauswirth@unil.ch

Peter F. Sweeney
IBM Research
Hawthorne, NY, USA
pfsweeney@us.ibm.com
Focus efforts on replication
Eval as improvement to the science

The value of replicates…

Specify as a causal graphical model

Evaluators produce replicates
Focus efforts on replication
Eval as improvement to the science
The value of replicates…
Specify as a causal graphical model

X has epistemic uncertainty
X has epistemic and aleatory uncertainty

Producing Wrong Data Without Doing Anything Obviously Wrong!

Todd Mytkowicz  Amer Diwan  Matthias Hauswirth  Peter F. Sweeney
Department of Computer Science  Amer Diwan  Faculty of Informatics  IBM Research
University of Colorado  University of Lugano  University of Lugano  Hawthorne, NY, USA
Boulder, CO, USA  Lugano, CH  Matthias.Hauswirth@unis.ch  pfsweeney.ibm.com
(mytkowicz@colorado.edu)
Focus efforts on replication
Eval as improvement to the science
The value of replicates...
Specify as a causal graphical model
Focus efforts on replication
Eval as improvement to the science
The value of replicates...
Specify as a causal graphical model

X has epistemic uncertainty
X has epistemic and aleatory uncertainty

Belief
O
Y
P
X
Y'
X'

Belief
O
Y
P
X
Y'
X'

Producing Wrong Data Without Doing Anything Obviously Wrong!

Todd Mytkowicz  Amer Diwan
Department of Computer Science
University of Colorado
Boulder, CO, USA
(mytkowicz@colorado.edu)

Matthias Hauswirth
Faculty of Informatics
University of Lugano
Lugano, CH
Matthias.Hauswirth@unil.ch

Peter F. Sweeney
IBM Research
Hawthorne, NY, USA
psf@us.ibm.com
Focus efforts on replication
Eval as improvement to the science

The value of replicates...

Specify as a causal graphical model

Elements of M chosen arbitrarily (good enough)

Producing Wrong Data Without Doing Anything Obviously Wrong!

Evaluator produce replicates
Outline

Oral History of Artifact Evaluation (student perspective)

Evaluators produce replicates

Language design for reproducibility
Why this is interesting
CGMs are hard to get right

- Abuse of plate notation?
  - $Y'$ is \textit{not} randomly sampled
  - Should $X'$ be a random variable?
  - Should we have a separate value for $P$?

Language design for reproducibility
Better: state assumptions in a language
Specifically, a hypothesis language

```
O : { "02", "03" }
Y : nat
(progid) Y <- 0
sharp (progid) assert (Y > 0)
Y_A = Y | O = "02"
Y_B = Y | O = "03"
(progid) assert (Y_A > Y_B)
```

```
for trialid in repeat(progid, machineid) under complete:
    measure(Y)
```
Better: state assumptions in a language

Specifically, a hypothesis language

```plaintext
O : { "02", "03" }
Y : nat
E : nat
P : { "Pentium4", "Core2", "m503CPU" }
C : { "gcc", "intel" }
L : nat
(progid) Y <- O, L, E, C, P
Y_A = Y | O = "02", L
Y_B = Y | O = "03", L
(progid) assert (Y_A > Y_B)
(progid) Y_B >>= E
```
HyPL

\[
\begin{align*}
\text{op} & ::= = | > | < \\
\text{coef} & ::= ? | n \\
\text{sup} & ::= \text{nat} | \text{bool} | \{\text{str}_1, \text{str}_2, \ldots, \text{str}_n\} | \text{real} \\
\text{decl} & ::= X : \langle\text{sup}\rangle | X : \langle\text{sup}\rangle \text{ of } (\text{unitid}_i) | Y' = Y | (X_1 \langle\text{op}\rangle v_1, \ldots, X_n \langle\text{op}\rangle v_n) \\
\text{hfn} & ::= \langle\text{coef}\rangle | \langle\text{coef}\rangle X | \langle\text{coef}\rangle X_1 X_2 | \langle\text{coef}\rangle \exp(\langle\text{hfn}\rangle) | \langle\text{hfn}\rangle + \langle\text{hfn}\rangle \\
\text{htype} & ::= \text{sharp} (\text{unitid}_i) | (\text{unitid}_i) | \text{belief} \\
\text{bexp} & ::= \top | \bot | X | ! \langle\text{bexp}\rangle | \langle\text{bexp}\rangle \&\& \langle\text{bexp}\rangle | \langle\text{bexp}\rangle \| \langle\text{bexp}\rangle | \langle\text{hfn}\rangle \langle\text{op}\rangle \langle\text{hfn}\rangle | \langle\text{hyp}\rangle \\
\text{hyp} & ::= \langle\text{htype}\rangle Y := \langle\text{hfn}\rangle | \langle\text{htype}\rangle Y \leftarrow X | \langle\text{htype}\rangle Y \leftarrow X|Z | \langle\text{htype}\rangle Y \implies X \\
\text{stmt} & ::= \langle\text{decl}\rangle | \langle\text{hyp}\rangle \\
\text{model} & ::= \langle\text{stmt}\rangle \\
\end{align*}
\]

Language design for reproducibility
Why another PPL?

It’s not all about the parameters
Additional affordances via language-based approach
Enables: Structured Search
...or, search beyond keywords

1  O : { "02", "03" }
2  Y : nat
3  E : nat
4  P : { "Pentium4", "Core2", "m503CPU" }
5  C : { "gcc", "intel" }
6  L : nat
7  (progid) Y <- O, L, E, C, P
8  Y_A = Y | O = "02", L
9  Y_B = Y | O = "03", L
10 (progid) assert (Y_A > Y_B)
11 (progid) Y_B --> E

Language design for reproducibility
Enables: Continuous Auditing

...or, regression testing for past studies

Language design for reproducibility
Enables: Onboarding neophytes
Make adhering to best practices easier!

SIGPLAN Empirical Evaluation Checklist

This checklist is meant to support informed judgement, not supplant it.

- claims not explicit
  - claims must be explicit in order for the reader to assess whether the empirical evaluation supports them. Missing claims cannot possibly be assessed. Claims should also aim to state not just what is achieved but how.

- relevant metrics
  - indirect or inappropriate proxy metric
    - proxy metrics can substitute for direct ones only when the substitution is clearly, explicitly justified. For example, it would be misleading and incorrect to report a reduction in cache misses to claim actual end-to-end performance or energy consumption improvement.

- fails to measure at important effects
  - all important effects should be measured to show the true cost of a system. For example, compiler optimizations may speed up programs at the cost of drastically increasing compile times of large systems, so the compile time should be measured as well as the program speedup. Failure to do so distorts the cost/benefit of the system.

- fails to acknowledge limitations
  - a paper should acknowledge its limitations to place the scope of its results in context. Stating no limitations at all, or only tangential ones, while omitting the more relevant ones may mislead the reader into drawing overly-strong conclusions. This could hold back efforts to publish future improvements, and may lead researchers down wrong paths.

- fails to compare against appropriate baseline
  - empirical evidence for a claim that a technique/system improves upon the state-of-the-art should include a comparison against a baseline. The lack of a baseline means empirical evidence lacks context. A straw man baseline that is misrepresented as state-of-the-art is also problematic, as it would inflate apparent benefit.

- comparison is unfair
  - comparisons to a competing system should not unfairly dis-advantage that system. Doing so may disfigure the apparent benefit.

Language design for reproducibility
Challenges in application to cybersecurity
Extreme values

Interested in maxima or the long tail?

Need different methods!
Observe Phenomenon
- Form Hypothesis
- Run Experiment
- Measure Results
- Analyze Results
- Draw Conclusion
- Publish

Extreme values & Non-scientific knowledge
Not an end, but hopefully a