Dare to Share: Risks and Rewards of Artifact Sharing in Computer Science

ACSAC 2017

Christian Collberg 
Todd Proebsting
Department of Computer Science
University of Arizona

http://repeatability.cs.arizona.edu
http://findresearch.org

Supported by the private foundation that must not be named
Opening Gambit
The Deception Study
8 Artifact Sharing Proposals
8 Laws of Artifact Sharing
Risks and Rewards
Some Computer Security Paper

Well-known Authors

Abstract

We present a new general technique for protecting clients in distributed systems against software misbehavior and (R-MATE) attacks. Such attacks occur in settings where an adversary has physical access to an intact client device and can obtain an advantage from tampering with the hardware itself or the software it contains.

In our system, the trusted server administers the attained client’s authorized abilities by continuously and autonomously generating and testing bit-level diverse client code variants. The diversity submodel employs a set of primitive code transformations that provide an ever-changing attack target for the adversary, making tampering difficult without being detected by the service.

1. Introduction

After-analyzed (R-MATE) attacks occur in settings where an adversary has physical access to a device and can tamper with its hardware or software. Remote manipula-
tion (R-MATE) attack occurs in distributed systems where an attacker can in frequent communication with several servers over a network, and malicious user cadet advantage is computed on an external device. To illustrate the adequacy of R-MATE manipulations, consider the following four situations. First, in the adversarial manipula-
tion framework (R-MATE) for controlling the adversarial protocol, an adversarial process (R-MATE) is initiated at a location in the network. This process initiates a command to a server or a client. The server or client then interacts with the server or client to execute an additional process, or to take a control action to affect communication between components [44]. Second, malicious modifies or replaces a susceptible in R-
MATE attacks. A notable example of this type is a server or client who tampers with the generation of a request against other policies [44]. Third, resistive measures are often deployed to assess an adversary’s capabilities (e.g., where they are connected to or tampered with). A compartmental network can be occluded into applying the tampering observations to a baseline, causing system damage. Finally, while current resilience mechanisms are typically protected by encryption while stored in distributed and/or remote servers’ offices, they are vulnerable to R-MATE attacks if an individual server’s computer network is compromised.

1.1 Overview

In each of the scenarios above, an attacker’s goal is to tamper with the client code and data under the control. The trusted server’s goal is to detect and mitigate events, which might compromise such a server (whether local or remote).

Security mechanisms, in this paper we present a system that achieves protection against R-MATE attacks through the use of code diversity and continuous code replacement. In our system, the internal server continuously and automatically generates diverse versions of client code and delivers updates to the attacked clients. For a long time, the client is continuously analyzed and updated to ensure code remains, thereby making tampering difficult without being detected by the service.

The client is designed to continuously change code without being detected by the adversary.

Limitations. Our system is specifically targeted distributed applications which have frequent client-server communication, where client tampering can only be internal at client-server in-

trusion events. Furthermore, while our use of code diversity can delay our attacks, it cannot necessarily prevent it. One goal is therefore the rapid detection of attack activities which need to completely prevent any tampering of client code. For now, the threat to the client is large, or a considerable length for our system. To do this, consider the following timeline in the history of adversarial motivations concerning our system:

This is an overview of events, point in time when chunks communicate with servers to either make copies of the product code updates. At time t, the client tamper with the code under his control. Until the next intrusion event, neither the client nor server notices the attack. At some point, after an interval t, consisting of a few interaction periods, the client’s tampering has caused the display anomalous behavior, perhaps through the use of the advan-
tageous manipulation protocol, with the server across time. As time progresses, finally, the server detects suspicious patterns by listening.
Some Computer Security Paper

Authors

Abstract

We present a distributed attack method that exploits a
physical access to a device, likely the hard drive.

In our system, the trusted root computer
client's administrative ability by continuous,
generating a new root key to resist attacks.

Many of the computer systems have been
been compromised using this technique,
making tamper-free attacks difficult without
this in the future.

1. Introduction

The method described above involves an
intrusion into a device that has a physical
access to a device. By tampering with the
hardware or software, a root exploit can be
achieved. An attacker can then install a
root exploit on the device to gain access.

To illustrate the simplicity of our
method, consider the following example:

A user accesses a device on the network
via a terminal. The device has a root exploit
installed, which allows the attacker to
gain access to the device. By using this
exploit, the attacker can then install a
root exploit on the device to gain access.

This allows the attacker to gain access
to the device and install a root exploit
on the device to gain access.

As can be seen, this method is
simple and effective. It allows the
attacker to gain access to a device
without having to tamper with the
hardware or software.
To: authors@cs.ux.edu

Cool paper! Can you send me your system so I can break it? 😊

Thanks!

Christian
To: authors@cs.ux.edu

Cool paper! Can you send me your system so I can break it? 😊

Thanks!
Christian
Reimplement!

define type operator =
    A
    B of operand * value * binop
    C of operand * value * operand * binop
    D of operand * value * operand * binop
    E of operand * operand
To: authors@cs.ux.edu
f: never used!
g: not defined!
h: doesn’t type check!
i: different in TR and paper!
I ... have few recollections of the work. [It was] like seeing a new paper for the first time.
To: PI,DC@cs.ux.edu

Request under the OPEN RECORDS ACT ... ALL RESEARCH ARTIFACTS ...
From: legal@cs.ux.edu

... to the extent such records may exist, they will not be produced pursuant to ORA.
From: legal@cs.ux.edu

... and no, they don’t exist...
Really?

PhD Thesis

10 TOP

NSF

NSF

DARPA
Pursuant to ORA, I request copies of all electronic mail...
... a total cost of $2,263.66 to search for, retrieve, redact and produce such records.
We will also make our data and software available to the research community when appropriate.
Consequences

By not sharing their artifacts, and by (perhaps unintentionally) leaving holes in their publications, the authors have effectively guaranteed that their claims can never be refuted.
Consequences

By not sharing their artifacts, and by (perhaps unintentionally) leaving holes in their publications, the authors have effectively guaranteed that their claims can never be refuted.
8th Law of Artifact Sharing
(Pretschner’s Law)

The probability of getting code out of someone is inversely proportional to the outrageousness of the claims in the paper.
Research Artifacts

- Code
- Data
- Experiments
- ...

{ }

Graduation cap icon
Research Artifacts

- Code
- Data
- Experiments
- ...

Repeatability

Verify results

Research Artifacts
Research Artifacts

- Code
- Data
- Experiments
- ...

Reproducibility

- New Experiment

Verify results

Repeatability

Research Artifacts

Confirm Hypothesis
The Deception Study
601 Research Papers
601 Research Papers

Has code?
Has code?

Can we find it?

1. Article?
2. Web?
3. Email?
601 Research Papers

Has code?

Can we find it?

1. Article?
2. Web?
3. Email?

1. ≤30 mins?
2. >30 mins?
3. Author?

Does it “work”?
601 Research Papers

Has code?

Can we find it?

Does it “work”?

1. Article?
2. Web?
3. Email?

1. ≤30 mins?
2. >30 mins?
3. Author?

Weakly Repeatable
Authors share their code, and it builds.
NC 63  HW 30

Article 85  Web 54  EM$^{yes}$ 87

EX 106  EM$^{g}$ 30  EM$^{no}$ 146

OK$^{≤30}$ 130  OK$^{>30}$ 64  OK$^{Auth}$ 23

Fails 9
The good news ... I was able to find some code. I am just hoping that it ... matches the implementation we ... used for the paper.
Unfortunately the current system is not mature ... We are actively working on a number of extensions ... Soon ...
[Our] prototype ... included many moving pieces that only [student] knew how to operate ... he left.
[Our] prototype ... included many moving pieces that only [student] knew how to operate ... he left.
... the server in which my implementation was stored had a disk crash ... three disks crashed ... Sorry for that.
... the server in which my implementation was stored had a **disk crash** ... three disks crashed ... Sorry for that.
The code ... is ... **hardly usable by anyone** other than the authors ... due to our decision to use [obscure variant of obscure language]
7th Law of Artifact Sharing

Unless a project starts with the express goal of post-publication artifact sharing, getting the right code, in a timely fashion, out of the project is virtually impossible.
We will not provide the software ... [because we spent] more time getting outsiders up to speed than on our own research.
... we can't share what did for this paper. ... this is not in the academic tradition, but this is a hazard in an industrial lab.
We have no plans to make the scheduler's source code publicly available ... because [ancient OS] as such does not exist anymore.
Based on bad experiences, we don’t want our system used in situations that it wasn’t meant for. Tell us if you use it to publish comparisons with other techniques.
We have an agreement with the [business], and we cannot release the code because of the potential privacy risks ...
Available Soon...

Personnel

Academic Pressure

Sw/Hw

Obsolet Sw/Hw

Don’t want

Fear

Licensing

Fear

Industrial Lab Issues

Privacy/Security

Poor Design
Sharing Proposal
— #1 —
Artifact Curation
Sharing Proposal - #1 -

Artifact Curation

TECHNO-FLIP
Refactoring Java Generics by Inferring Wildcards, In Practice

John Altidor  
University of Massachusetts  
jaltidor@cs.umass.edu

Yannis Smaragdakis  
University of Athens  
smaragd@di.uoa.gr

APPENDICES and SUPPLEMENTS

artifact_overview.pdf (100 KB)  Artifact Overview for Paper #35 of OOPSLA 2014
oopsla035.zip (95.77 MB)  Please, email questions to jaltidor@cs.umass.edu
VarJ.zip (95.77 MB)  Please, email questions to jaltidor@cs.umass.edu
RunMyCode enables scientists to openly share the code and data that underlie their research publications.

This service is based on the innovative concept of a companion website associated with a scientific publication.
## FindResearch.org

1. Help the public find artifacts
2. Motivate researchers to share

<table>
<thead>
<tr>
<th>Title/Authors</th>
<th>Research Artifacts</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal inference of fields in row-polymorphic records</td>
<td>Axel Simon</td>
<td>Discussion Comments: 0</td>
</tr>
<tr>
<td>VeriCon: towards verifying controller programs in software-defined networks</td>
<td>Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shishar Izhaky, Aleksandr Karbyshev, Moody Sagiv, Michael Schapira, Asaf Valdsterisky</td>
<td><a href="http://www.cs.tau.ac.il/~shishar">http://www.cs.tau.ac.il/~shishar</a></td>
</tr>
<tr>
<td>Tracelet-based code search in executables</td>
<td>Yaniv David, Enn Yehav</td>
<td><a href="https://github.com/Yanivmd/TRACY">https://github.com/Yanivmd/TRACY</a></td>
</tr>
<tr>
<td>Modular control-flow integrity</td>
<td>Ben Niu, Gang Tan</td>
<td>Discussion Comments: 0</td>
</tr>
<tr>
<td>Laws of concurrent programming</td>
<td>Tony Hoare</td>
<td>Discussion Comments: 0</td>
</tr>
<tr>
<td>Test-driven repair of data races in structured parallel programs</td>
<td>Ravi Sunkavalli, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar</td>
<td><a href="http://dl.acm.org/ct_gateway.cfm?id=29943">http://dl.acm.org/ct_gateway.cfm?id=29943</a></td>
</tr>
<tr>
<td>Title/Authors</td>
<td>Research Artifacts</td>
<td>Details</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------</td>
<td>--------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Optimal inference of fields in row-polymorphic records</td>
<td></td>
<td>Discussion Comments: 0, Verification: Author has not verified information, More...</td>
</tr>
<tr>
<td>Axel Simon</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VeriCon: towards verifying controller programs in software-defined networks</td>
<td>• <a href="http://www.cs.tau.ac.il/~shachar">http://www.cs.tau.ac.il/~shachar</a></td>
<td>Discussion Comments: 0, Verification: Authors have not verified information, More...</td>
</tr>
<tr>
<td>Thomas Ball, Nikolaaj Bjørner, Aaron Gember, Shachar Itzhaky,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, AsafVoisardsky</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tracelet-based code search in executables</td>
<td>• <a href="https://github.com/Yanivmd/TRACY">https://github.com/Yanivmd/TRACY</a></td>
<td>Discussion Comments: 0, Verification: Authors have not verified information, More...</td>
</tr>
<tr>
<td>Yaniv David, Enen Yahav</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modular control-flow integrity</td>
<td></td>
<td>Discussion Comments: 0, Verification: Authors have not verified information, More...</td>
</tr>
<tr>
<td>Ben Niu, Gang Tan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dappio: breaking the browser language barrier</td>
<td>• <a href="http://www.dappiium.org/">http://www.dappiium.org/</a></td>
<td>Discussion Comments: 0, Verification: Authors have not verified information, More...</td>
</tr>
<tr>
<td>John Vlk, Einery D. Berger</td>
<td>• Artifacted evaluation badge awarded</td>
<td></td>
</tr>
<tr>
<td>Laws of concurrent programming</td>
<td></td>
<td>Discussion Comments: 0, Verification: Author has not verified information, More...</td>
</tr>
<tr>
<td>Tony Hoare</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test-driven repair of data races in structured parallel programs</td>
<td>• <a href="http://dl.acm.org/ft_gateway.cfm?id=259437">http://dl.acm.org/ft_gateway.cfm?id=259437</a></td>
<td>Discussion Comments: 0, Verification: Authors have not verified information, More...</td>
</tr>
<tr>
<td>Rath Suresh, Raghavan Raman, Swarit Chaudhuri, John M. Melter-Crummey, Vivek Sarkar</td>
<td>• Artifacted evaluation badge awarded</td>
<td></td>
</tr>
</tbody>
</table>
# ACM Programming Language Design and Implementation, PLDI 2014

<table>
<thead>
<tr>
<th>Title/Authors</th>
<th>Research Artifacts</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal inference of fields in row-polymorphic records</td>
<td></td>
<td>Discussion Comments: 0  Verification: Author has not verified information  More...</td>
</tr>
<tr>
<td>Axel Simon</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VeriCon: towards verifying controller programs in software-defined networks</td>
<td><a href="http://www.cs.tau.ac.il/~shachar">http://www.cs.tau.ac.il/~shachar</a></td>
<td>Discussion Comments: 0  Verification: Authors have not verified information  More...</td>
</tr>
<tr>
<td>Thomas Ball, Nikola Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Moshe Sagiv, Michael Schapira, Assaf Vitalis</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tracelet-based code search in executables</td>
<td><a href="https://github.com/Yanivmd/TRACY">https://github.com/Yanivmd/TRACY</a></td>
<td>Discussion Comments: 0  Verification: Authors have not verified information  More...</td>
</tr>
<tr>
<td>Yaniv David, Enni Yahav</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modular control-flow integrity</td>
<td></td>
<td>Discussion Comments: 0  Verification: Authors have not verified information  More...</td>
</tr>
<tr>
<td>Ben Niu, Gang Tan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dappio: breaking the browser language barrier</td>
<td><a href="http://www.dappiojm.org/">http://www.dappiojm.org/</a></td>
<td>Discussion Comments: 0  Verification: Authors have not verified information  More...</td>
</tr>
<tr>
<td>John Vilk, Einar D. Berger</td>
<td>Artifacts evaluation badge awarded</td>
<td></td>
</tr>
<tr>
<td>Laws of concurrent programming</td>
<td></td>
<td>Discussion Comments: 0  Verification: Author has not verified information  More...</td>
</tr>
<tr>
<td>Tony Hoare</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test-driven repair of data races in structured parallel programs</td>
<td><a href="http://dl.acm.org/ft_gateway.cfm?id=23943">http://dl.acm.org/ft_gateway.cfm?id=23943</a></td>
<td>Discussion Comments: 0  Verification: Authors have not verified information  More...</td>
</tr>
<tr>
<td>Rath Suresh, Raghavan Raman, Swarukta Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar</td>
<td>Artifacts evaluation badge awarded</td>
<td></td>
</tr>
</tbody>
</table>
FindArtifacts

Verify

Information

FindArtifacts

Emails, Grants

Verify

Information

FindResearch.org

ACM Programming Language Design and Implementation, PLDI 2014

<table>
<thead>
<tr>
<th>Title/Authors</th>
<th>Research Artifacts</th>
<th>Details</th>
</tr>
</thead>
</table>
| Optimal inference of fields in row-polyorphic records | | Discussion Comments: 0
Verification: Author has not verified information More... |
| Axel Simon | | |
| VeriCon: towards verifying controller programs in software-defined networks | * http://www.cs.tau.ac.il/~shachar | Discussion Comments: 0
Verification: Authors have not verified informat... More... |
| Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shuschar Itzhaky, Aleksandr Karbyshev, Moody Sagiv, Michael Schapira, Asaf Valdarsky | | |
| Tracelet-based code search in executables | * https://github.com/Yanivmd/TRACY | Discussion Comments: 0
Verification: Authors have not verified informat... More... |
| Yaniv David, Enri Yahav | | |
| Modular control-flow integrity | | Discussion Comments: 0
Verification: Authors have not verified informat... More... |
| Ben Niu, Gang Tan | | |
| Dauplo: breaking the browser language barrier | * http://www.dauplojvm.org/ | Discussion Comments: 0
Verification: Authors have not verified informat... More... |
| John Vlk, Enny D. Berger | * Artifactual evaluation badge awarded | |
| Laws of concurrent programming | | Discussion Comments: 0
Verification: Author has not verified information More... |
| Tony Hoare | | |
| Test-driven repair of data races in structured parallel programs | * http://dl.acm.org/f_gateway.cfm?id=25943 | Discussion Comments: 0
Verification: Authors have not verified informat... More... |
| Rith Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar | * Artifactual evaluation badge awarded | |
FindResearch.org

- Find Artifacts
- Emails, Grants
- Verify
- Information
- Publish

**ACM Programming Language Design and Implementation, PLDI 2014**

<table>
<thead>
<tr>
<th>Title/Authors</th>
<th>Research Artifacts</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal inference of fields in row-polyomorphic records</td>
<td>Axel Simon</td>
<td>Discussion Comments: 0 Verification: Author has not verified information More...</td>
</tr>
<tr>
<td>VeriCon: towards verifying controller programs in software-defined networks</td>
<td>Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karpyshev, Moody Sagiv, Michael Schapira, Asaf Valdsterly</td>
<td><a href="http://www.cs.tau.ac.il/~shachar">http://www.cs.tau.ac.il/~shachar</a> Verification: Authors have not verified information More...</td>
</tr>
<tr>
<td>Tracelet-based code search in executables</td>
<td>Yaniv David, Enen Yahav</td>
<td><a href="https://github.com/Yanivmd/TRACEY">https://github.com/Yanivmd/TRACEY</a> Verification: Authors have not verified information More...</td>
</tr>
<tr>
<td>Modular control-flow integrity</td>
<td>Ben Niu, Gang Tan</td>
<td>Discussion Comments: 0 Verification: Author has not verified information More...</td>
</tr>
<tr>
<td>Dappio: breaking the browser language barrier</td>
<td>John Wilk, Einy E. Berger</td>
<td><a href="http://www.dappijum.org/">http://www.dappijum.org/</a> Verification: Authors have not verified information More...</td>
</tr>
<tr>
<td>Laws of concurrent programming</td>
<td>Tony Hoare</td>
<td>Artificial evaluation badge awarded</td>
</tr>
<tr>
<td>Test-driven repair of data races in structured parallel programs</td>
<td>Rohit Sureshnan, Raghavaran Ramani, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar</td>
<td><a href="http://dl.acm.org/ft_gateway.cfm?id=25943-">http://dl.acm.org/ft_gateway.cfm?id=25943-</a> Verification: Authors have not verified information More...</td>
</tr>
</tbody>
</table>
Find Research.org

Find Artifacts
- Emails, Grants

Verify Information

Publish

Discuss!
- 120 conferences
- 12,000 articles
- 47,000 unique authors
- 43,000 verification emails sent
• 120 conferences
• 12,000 articles
• 47,000 unique authors
• 43,000 verification emails sent
• 9% of articles are verified
• 6% of articles have shared artifacts
Papers/Year (Dblp)
6th Law of Artifact Sharing

Even if you built it, they still won’t come.
Does it work? (Repeatability)

Why do you want my code?
Will the code help me understand the paper?

Why do you want my code?
Can I build on it? (Benefaction)

Why do you want my code?
How does it compare to my work?

Why do you want my code?
Does it reproduce?

Why do you want my code?
It’s on GitHub! I’m done!
Share everything
Where’s abclib.so?

Ensure longevity: include all external code

Just apt-get it! Works for me!
Which gcc version???

Uhm, 4.2 or higher?

☑ Document software you can’t include
Uhm, I think this one?

Clearly link paper to artifact
Uhm, I think this one?

The paper should clearly link to the artifact.

Paper $\Rightarrow$ github!
Where is your artifact?

Ensure availability: find permanent storage
Where is your artifact?

3% of verified papers with shared artifacts have broken links

Ensure availability: find permanent storage
Can you help me?

Use permanent email addresses
Can you help me?

- 9% of emails bounced
- Many articles without any email address

✔️ Use permanent email addresses
5th Law of Artifact Sharing

To ensure repeatability of your results by others, you must
1. share everything
2. assume nothing
3. remain reachable
Sharing Proposal
— #3 —

Tool Support
Executable Paper 1

Abstract

We present a new general technique for protecting client systems against remote code execution (RCE) attacks. Our approach involves two main steps: first, the client is modified to include a "tryme" functionality, and second, the modified client is used to establish a secure channel for communicating with the server.

1. Introduction

1.1 Overview

In order to achieve the goals set in the previous section, the "tryme" functionality is employed to establish a secure channel for communicating with the server. This channel is used to transmit commands and responses, without the risk of interception by third parties.
1. Introduction

1.1 Overview

In 1996 the government of the Netherlands introduced the Secure Payment System (CPS) to improve the security of financial transactions. This system allows customers to make payments online by using a smart card and a PIN. The main advantage of the CPS is that it provides a secure and convenient way to make payments over the internet. However, the CPS also raises some security concerns, especially in terms of the confidentiality and integrity of the transactions. To address these concerns, we propose a new model for secure online payments that is based on a combination of traditional and modern cryptographic techniques.

The new model is designed to provide a high level of security while maintaining the convenience of the CPS. It uses a combination of encryption and digital signatures to ensure the confidentiality and integrity of the transactions. The model also includes a mechanism for verifying the identity of the parties involved in the transaction, which helps to prevent fraud.

The proposed model is implemented in a set of executable papers, which are available on the website is.ieis.tue.nl/staff/pvgorp/share. The papers are written in LaTeX and include all the necessary code and experiments to demonstrate the feasibility of the new model. The experiments were conducted using the tryme framework, which is available at the same website.

To summarize, the new model for secure online payments is designed to provide a high level of security while maintaining the convenience of the CPS. It uses a combination of traditional and modern cryptographic techniques and includes a mechanism for verifying the identity of the parties involved in the transaction. The model is implemented in a set of executable papers, which are available on the website is.ieis.tue.nl/staff/pvgorp/share. The papers include all the necessary code and experiments to demonstrate the feasibility of the new model.
Executable Paper 1

is.ieis.tue.nl/staff/pvgorp/share
VisTrails

Workflow v1.0

Data

Paper

tryme
VisTrails

Workflow v1.0

- Input
- R
- Python
- Plot

Data

Paper

Abstract

1.1 Overview

tryme
VisTrails

Workflow v1.1

Workflow v1.0

Input

R

Python

Plot

Data

Paper

tryme

Abstract

We present a new approach to developing, sharing, and executing scientific workflows. We describe the workflow development and execution infrastructure that supports this approach, and we demonstrate its utility with a case study.

1. Introduction

In this work, we address the problem of developing and executing scientifi c workflows. We describe a new workflow development and execution infrastructure that supports this approach, and we demonstrate its utility with a case study.
ReproZip-Pack

> task1.py
> task2.py

open()
exec()
ReproZip-Pack

> task1.py
> task2.py

open()
exec()

trace

python
libc
task1

www.reprozip.org
ReproZip-Pack

> task1.py
> task2.py

TRACE

open()
exec()

ReproZip-Unpack

> run task1
> run task2

www.reprozip.org
4th Law of Artifact Sharing

When a Computer Scientist is first made aware of the Reproducibility Problem, their first thought is
4th Law of Artifact Sharing

When a Computer Scientist is first made aware of the Reproducibility Problem, their first thought is

Oh, I can build a tool for that!
Sharing Proposal
—#4—
Rewarding Good Behavior
Sharing Proposal

#4

Rewarding Good Behavior
PLDI 2017 PLDI Research Artifacts

Call for Research Artifacts

PLDI 2017 is continuing the novel experiment that began at PLDI 2014 and which has been employed for PLDI 2015 and 2016: giving authors the opportunity to submit for evaluation any artifacts that accompany their papers. Similar experiments ran successfully for OOPSLA, POPL, ESEC/FSE and ECOOP.

Background

A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to...
Paper accepted?

PLDI 2017 PLDI Research Artifacts

Call for Research Artifacts

PLDI 2017 is continuing the novel experiment that began at PLDI 2014 and which has been employed for PLDI 2015 and 2016: giving authors the opportunity to submit for evaluation any artifacts that accompany their papers. Similar experiments ran successfully for OOPSLA, POPL, ESEC/FSE, and ECOOP.

Background

A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to
Paper accepted?

Artifact accepted?

OK!
- Voluntary
- Does not affect accept/reject
- No expectation of sharing
Repeatability
Repeatability

Reproducibility

Benefaction
Accepted Artifacts / Accepted Papers (%)
# Conferences with AE

![Graph showing the number of conferences with AE from 2008 to 2017. The number of conferences increases steadily from 2008 to 2012, then sharply increases from 2012 to 2014, peaks in 2015, and remains relatively stable until 2017.](image-url)
Publication Venues (Dblp)
Sharing Proposal
— #5 —

Punishing Bad Behavior
We will make our data and software available.

Grant application #: xxxxxxxx
Random Audit!

Are you sharing like you promised in the grant application?
Sharing Contract!

What level of sharing are you committing to?
Sharing Contract

- License: ...
- Artifacts: source code, data, ...
- Where: ...
- Support: ...

Author

acm
Sharing Contract

- License: ...
- Artifacts: source code, data, ...
- Where: ...
- Support: ...

Accept/Reject?
Sharing Contract

- License: ...
- Artifacts: source code, data, ...
- Where: ...
- Support: ...

You promised!
Sharing Contract

- License: ...
- Artifacts: source code, data, ...
- Where: ...
- Support: ...

You promised!
Sharing Proposal
— #6 —
Optional Sharing
Sharing Proposal

#6

Mandated Sharing
Artifacts submitted with paper?
Artifacts submitted with paper?

Yes

Artifacts permit replication?

Yes
Artifacts submitted with paper?

Yes

Artifacts permit replication?

Review paper
Scientist
Sharing Proposal

--- #7 ---

Education
CS Research Methods Courses?
CS Research Methods Courses?

- Reading, writing, presenting, reviewing papers
- Experimental design
- Statistics, data processing, visualization
- Proposal writing, career issues
- Intellectual property, research ethics
CS Research Methods Courses?

- Reading, writing, presenting, reviewing papers
- Experimental design
- Statistics, data processing, visualization
- Proposal writing, career issues
- Intellectual property, research ethics

Reproducibility???
Keeping a complete and accurate record of experimental methods and data ... could someone else, ... use your notebook to repeat your work, and obtain the same results?
Reproducibility PI Manifesto

I pledge to

- teach grad students about reproducibility
- share artifacts at the time of submission
- add a *reproducibility statement* to papers

http://lorenabarba.com/gallery/reproducibility-pi-manifesto/
The dissertation proposal should state **if and how** they will provide access to code and data to support reproducibility.
Sharing Proposal — #8 —

All I Really Need to Know I Learned in Kindergarten
Why do we care about reproducibility and repeatability?
Why do we care about reproducibility and repeatability?
Why do we care about reproducibility and repeatability?
Why do we care about reproducibility and repeatability?
Dear B, I read your nice paper, thanks for sharing the code! However, I’m unable to reproduce your results.

Sincerely,

A
Dear A, thank you for pointing out our errors!

Best wishes,

B
Dear A, thank you for pointing out our errors!

Best wishes,

B
Are computer scientists hypercritical? Are we more critical than scientists and engineers in other disciplines? Bertrand Meyer’s August 22, 2011 The Nastiness Problem in Computer Science blog post partially makes the argument referring to secondhand information from the National Science Foundation (NSF). Here are some NSF numbers to back the claim that we are hypercritical.

This graph plots average reviewer ratings of all proposals submitted from 2005 to 2010 to NSF overall (red line), just Computer & Information Science & Engineering (CISE) (green line), and NSF minus CISE (blue line). Proposal ratings are based on a scale of 1 (poor) to 5 (excellent). For instance, in 2010, the average reviewer rating across all CISE programs is 2.96; all NSF directorates including CISE, 3.24; all NSF directorates excluding

CISE, 3.30.

https://cacm.acm.org/blogs/blog-cacm/134743-yes-computer-scientists-are-hypercritical/fulltext
Yes, Computer Scientists Are Hypercritical

By Jeannette M. Wing
October 6, 2011
Comments (15)

average reviewer rating across all CISE programs is 2.96; all NSF directorates including CISE, 3.24; all NSF directorates excluding CISE, 3.30.

https://cacm.acm.org/blogs/blog-cacm/134743-yes-computer-scientists-are-hypercritical/fulltext
SIGMOD 2010
350 submissions
Number of papers with all reviews “accept” or higher: 1
Are we malevolent grumps? Nothing personal, but as a community computer scientists sometimes seem to succumb to negativism. They admit it themselves. A common complaint in the profession is that instead of taking a cue from our colleagues in more cogently organized fields such as physics, who band together for funds, promotion, and recognition, we are incurably fractious. In committees, for example, we damage everyone’s chances by badmouthing colleagues with approaches other than ours. At least this is a widely perceived view ("Circling the wagons and shooting inward," as Greg Andrews put it in a recent discussion). Is it accurate?

One statistic that I have heard cited is that in 1-to-5 evaluations of projects submitted to the U.S. National Science Foundation the
Are we malevolent grumps?

... we damage everyone’s chances by badmouthing colleagues with approaches other than ours.

What Happened Next?

Submitted

Paper

Weakly Repeatable
Hate Us on Facebook!
Hate Us on Facebook!

Your site is violating IRB guidelines — take it down!
Hate Us on Facebook!

Your study stinks! Why didn’t you just...
Hate Us on Facebook!

Your students made rookie mistakes!
Hate Us on Facebook!

My code builds!
Hate Us on Facebook!

Fine it doesn’t build, but why didn’t you email me???
Turnabout is Fair Play!

http://cs.brown.edu/~sk/Memos/Examining-Reproducibility/
Please let us know if there's anything we can do in support of your efforts to examine our paper! We think your effort is terrific!
Notice

Please disregard the results below — they are included for completeness but contain numerous errors. We have redone the study and report the new results.

We originally put up this website so that the reviewers of our submitted paper could have access to our raw data, code, and technical report, in case they wished to review it. It was never publicly announced. Nevertheless, the site became public knowledge, and over the last week we have received many emails pointing out many apparent errors in the data. Some of these errors are, no doubt, a consequence of the definition of reproducibility we used in the study:

Can a CS student build the software within 30 minutes, including finding and installing any dependent software and libraries, and without bothering the authors?

As a result, we made another pass over the data, along with the people behind this site. We very much welcome these reviews-of-the-reviews - this is exactly the way science should work! Please don't hesitate to contact us should you have any further questions and comments.

This page is set to be unindexed by search engines.

Christian and Todd
Please let us know if there's anything we can do in support of your efforts to examine our paper! We think your effort is terrific!

A group of independent researchers set out to verify our build results through a crowdsourced effort; http://cs.brown.edu/~sk/Memos/Examining-Reproducibility

Acknowledgments
We would like to thank Saumya Debray, Shriram Krishnamurthi, Alex Warren, and the anonymous reviewers for valuable input.

Christian and Todd
Please let us know if there's anything we can do in support of your efforts to examine our paper!

We think your effort is terrific!
They did *crap* work, would not admit to when caught out and even pretended it hadn’t happened.
...these researchers have done a disservice to science by publishing a study they knew to be horse manure, and then piling more bull crap on it when caught ... they are simply trying to build a reputation off a problem they don't really care to solve ...
To the University of Arizona Institutional Review Board:

Revoke their IRB permission!
1. Their deception study was bad – I don’t trust them!
1. Their deception study was bad – I don’t trust them!
2. They’re violating my privacy!

The authors
- have
- have not verified
1. Their deception study was bad — I don’t trust them!
2. They’re violating my privacy!
3. They’re spying on my computer!
3rd Law of Artifact Sharing

Without a culture of respectful academic interchange, where failure is seen as an accepted part of the progression of science, sharing will not become default behavior.
Risks vs. Rewards
Licensing

Obsolete SW/HW

Academic Pressure

Versioning

Available

Industrial Lab Issues

Privacy/Security

Poor Design

Versioning

Academic Pressure

Industrial Lab Issues

Privacy/Security

Available

Obsolete SW/HW
## ACM Programming Language Design and Implementation, PLDI 2014

<table>
<thead>
<tr>
<th>Title/Authors</th>
<th>Research Artifacts</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal inference of fields in row-polymorphic records</td>
<td>Axel Simon</td>
<td>Discussion Comments: 0&lt;br&gt;Verification: Author has not verified information&lt;br&gt;More...</td>
</tr>
<tr>
<td>VeriCon: towards verifying controller programs in software-defined networks</td>
<td>Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Izhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valdrasovsky</td>
<td>Discussion Comments: 0&lt;br&gt;Verification: Authors have not verified information&lt;br&gt;More...</td>
</tr>
<tr>
<td>Tracelet-based code search in executables</td>
<td>Yaniv David, Eren Yahav</td>
<td>Discussion Comments: 0&lt;br&gt;Verification: Authors have not verified information&lt;br&gt;More...</td>
</tr>
<tr>
<td>Modular control-flow integrity</td>
<td>Ben Niu, Giang Tan</td>
<td>Discussion Comments: 0&lt;br&gt;Verification: Authors have not verified information&lt;br&gt;More...</td>
</tr>
<tr>
<td>Doppio: breaking the browser language barrier</td>
<td>John Vlk, Emery D. Berger</td>
<td>Discussion Comments: 0&lt;br&gt;Verification: Authors have not verified information&lt;br&gt;More...</td>
</tr>
<tr>
<td>Laws of concurrent programming</td>
<td>Tony Hoare</td>
<td>Discussion Comments: 0&lt;br&gt;Verification: Author has not verified information&lt;br&gt;More...</td>
</tr>
<tr>
<td>Test-driven repair of data races in structured parallel programs</td>
<td>Rishi Surendran, RagHAVan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar</td>
<td>Discussion Comments: 0&lt;br&gt;Verification: Authors have not verified information&lt;br&gt;More...</td>
</tr>
</tbody>
</table>

FindResearch.org

Find artifacts... Verify Publish
Find artifacts...
Verify
Publish

9.4% of articles are verified
Find artifacts...

Verify

Publish

9.4% of articles are verified
10.2% of articles are verified with $5 STARBUCKS CARD
<table>
<thead>
<tr>
<th>Rewards</th>
<th>Risks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Rewards</td>
<td>Risks</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td><strong>Credibility</strong>: Colleagues may trust your work more when they can try it.</td>
<td></td>
</tr>
<tr>
<td><strong>Rewards</strong></td>
<td><strong>Risks</strong></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
</tr>
<tr>
<td><strong>Credibility:</strong> Colleagues may trust your work more when they can try it.</td>
<td></td>
</tr>
<tr>
<td><strong>Visibility:</strong> Colleagues may notice your work when they can build on the code.</td>
<td></td>
</tr>
<tr>
<td>Rewards</td>
<td>Risks</td>
</tr>
<tr>
<td>------------------</td>
<td>-------</td>
</tr>
<tr>
<td><strong>Credibility</strong>:</td>
<td></td>
</tr>
<tr>
<td>Colleagues may</td>
<td></td>
</tr>
<tr>
<td>trust your work</td>
<td></td>
</tr>
<tr>
<td>more when they</td>
<td></td>
</tr>
<tr>
<td>can try it.</td>
<td></td>
</tr>
<tr>
<td><strong>Visibility</strong>:</td>
<td></td>
</tr>
<tr>
<td>Colleagues may</td>
<td></td>
</tr>
<tr>
<td>notice your work</td>
<td></td>
</tr>
<tr>
<td>when they can</td>
<td></td>
</tr>
<tr>
<td>build on the code.</td>
<td></td>
</tr>
<tr>
<td><strong>Correctness</strong>:</td>
<td></td>
</tr>
<tr>
<td>Colleagues may</td>
<td></td>
</tr>
<tr>
<td>help you find</td>
<td></td>
</tr>
<tr>
<td>and correct</td>
<td></td>
</tr>
<tr>
<td>problems.</td>
<td></td>
</tr>
<tr>
<td><strong>Rewards</strong></td>
<td><strong>Risks</strong></td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------</td>
</tr>
<tr>
<td><strong>Credibility:</strong> Colleagues may trust your work more when they can try it.</td>
<td></td>
</tr>
<tr>
<td><strong>Visibility:</strong> Colleagues may notice your work when they can build on the code.</td>
<td></td>
</tr>
<tr>
<td><strong>Correctness:</strong> Colleagues may help you find and correct problems.</td>
<td></td>
</tr>
<tr>
<td><strong>Collaboration:</strong> Colleagues may want to help you extend your system.</td>
<td></td>
</tr>
<tr>
<td>Rewards</td>
<td>Risks</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><strong>Credibility</strong>: Colleagues may trust your work more when they can try it.</td>
<td><strong>Credibility</strong>: Colleagues may find bugs and think your results are wrong.</td>
</tr>
<tr>
<td><strong>Visibility</strong>: Colleagues may notice your work when they can build on the code.</td>
<td></td>
</tr>
<tr>
<td><strong>Correctness</strong>: Colleagues may help you find and correct problems.</td>
<td></td>
</tr>
<tr>
<td><strong>Collaboration</strong>: Colleagues may want to help you extend your system.</td>
<td></td>
</tr>
<tr>
<td>Rewards</td>
<td>Risks</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td><strong>Credibility:</strong> Colleagues may trust your work more when they can try it.</td>
<td><strong>Credibility:</strong> Colleagues may find bugs and think your results are wrong.</td>
</tr>
<tr>
<td><strong>Visibility:</strong> Colleagues may notice your work when they can build on the code.</td>
<td><strong>Return-on-investment:</strong> Colleagues may ignore your code in spite of your efforts to share it.</td>
</tr>
<tr>
<td><strong>Correctness:</strong> Colleagues may help you find and correct problems.</td>
<td></td>
</tr>
<tr>
<td><strong>Collaboration:</strong> Colleagues may want to help you extend your system.</td>
<td></td>
</tr>
</tbody>
</table>
2nd Law of Artifact Sharing

The root of the scientific transparency problem is sociological, not technological: we do not share solid artifacts because there is little professional glory to be gained from doing so.
Can you believe, back in the 21st century, scientists would make up excuses why they shouldn’t have to share their research artifacts!
1st Law of Artifact Sharing
(Corollary to Max Planck’s Quip)

Scientific transparency advances one funeral at a time.
1. Propose a Research Methods course!
2. Help us populate FindResearch.org!
3. Prepare to share, use checklists!
4. Insist on Sharing Contracts!
Thank you!