# Software Integrity with Pegasus: Securing Scientific Workflow Data

November 14-15, 2017

SC17

CENTER FOR APPLIED CYBERSECURITY RESEARCH

Pegasus

SICE

renci

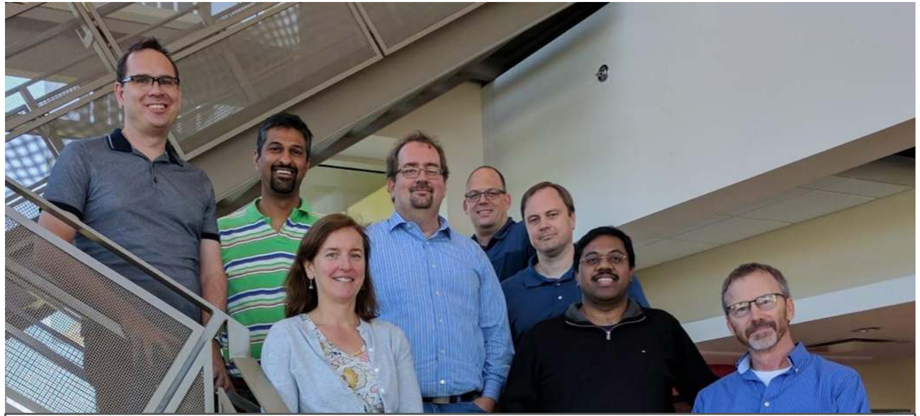
# Scientific Workflow Integrity with Pegasus
## NSF CICI Awards 1642070, 1642053, and 1642090

GOALS

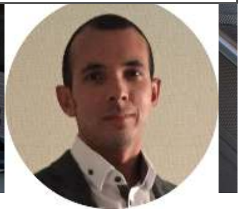
Provide additional assurances that a scientific workflow is not accidentally or maliciously tampered with during its execution

Allow for detection of modification to its data or executables at later dates to facilitate reproducibility.

Integrate cryptographic support for data integrity into the Pegasus Workflow Management System.



PIs: Von Welch, Ilya Baldin, Ewa Deelman, Steve Myers
Team: Omkar Bhide, Rafael Ferrieira da Silva, Randy Heiland, Anirban Mandal, Rajiv Mayani, Mats Rynge, Karan Vahi



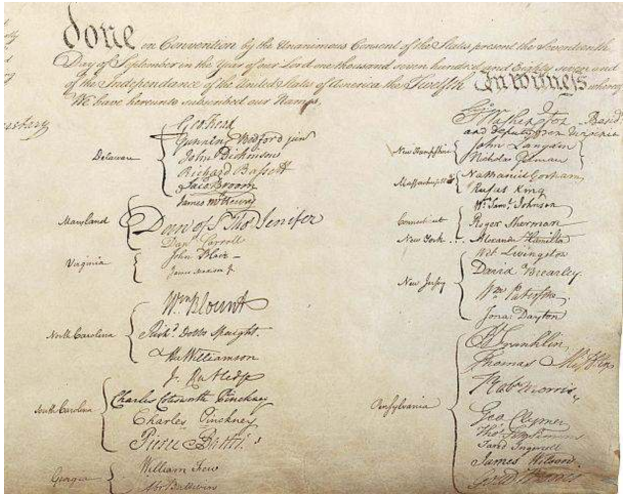
# Our Talk

- **Problem Statement: Challenges to Data Integrity**
- Our Approach: Adding integrity support to the popular Pegasus scientific workflow management system
- Challenges
- Next steps

# Data Integrity: Seal ~ Signature ~ Authenticity ~ Trust

Wax seal. Peng, CC BY-SA 3.0

Cylindrical seal http://www.ancient.eu/image/2833/ Osama Shukir Muhammed Amin, CC BY-SA 3.0


done in Convention by the Unanimous Consent of the States present the Seventeenth Day of September in the Year of our Lord one thousand seven hundred and Eighty seven and of the Independance of the United States of America the Twelfth In witness whereof We have hereunto subscribed our Names,

Go. Washington Presidt and deputy from Virginia

Delaware: Geo: Read, Gunning Bedford jun, John Dickinson, Richard Bassett, Jaco: Broom

Maryland: James McHenry, Dan of St Thos Jenifer, Danl Carroll

Virginia: John Blair, James Madison Jr.

North Carolina: Wm Blount, Richd Dobbs Spaight, Hu Williamson

South Carolina: J. Rutledge, Charles Cotesworth Pinckney, Charles Pinckney, Pierce Butler

Georgia: William Few, Abr Baldwin

New Hampshire: John Langdon, Nicholas Gilman

Massachusetts: Nathaniel Gorham, Rufus King

Connecticut: Wm Saml Johnson, Roger Sherman

New York: Alexander Hamilton

New Jersey: Wil: Livingston, David Brearley, Wm Paterson, Jona: Dayton

Pensylvania: B Franklin, Thomas Mifflin, Robt Morris, Geo. Clymer, Thos FitzSimons, Jared Ingersoll, James Wilson, Gouv Morris

Public Domain, https://commons.wikimedia.org/w/index.php?curid=662341


By Ipankonin - Vectorized from SVG elements from, Public Domain, https://commons.wikimedia.org/w/index.php?curid=1831846

Q: How does one “sign” digital data?

# Data Integrity

Important in Business, Arts, Politics, Science, ...

FAKE NEWS!

Media Digitization and Preservation

Reproducible Results

**Former NOAA Scientist Confirms Colleagues Manipulated Climate Records**

https://science.house.gov/news/press-releases/former-noaa-scientist-confirms-colleagues-manipulated-climate-records

vs.

***No Data Manipulation in 2015 Climate Study, Researchers Say***

https://www.nytimes.com/2017/02/07/science/2015-climate-study-data.html

# Our Focus: Science

## “Scientific Workflow Integrity with Pegasus”

Modern day [computational] science uses workflows extensively. One popular workflow management system (WMS) used by several NSF projects is Pegasus. A WMS allow scientists to describe their process in a human-friendly way and then the software handles the details of the processing, dealing with tedious and repetitive steps and handling errors.

https://pegasus.isi.edu/
https://github.com/pegasus-isi/pegasus

# Challenges to Scientific Data Integrity

Modern IT systems are not perfect - errors creep in.

At modern “Big Data” sizes we are starting to see checksums breaking down.

Plus there is the threat of intentional changes: malicious attackers, insider threats, etc.

# CERN Study of Disk Errors

Examined Disk, Memory, RAID 5 errors.

“The error rates are at the 10-7 level, but with complicated patterns.” E.g. 80% of disk errors were 64k regions of corruption.

Explored many fixes and their often significant performance trade-offs.

## Data integrity

Bernd Panzer-Steindel, CERN/IT
Draft 1.3 8. April 2007

### Executive Summary

We have established that low level data corruptions exist and that they have several origins. The error rates are at the $10^{-7}$ level, but with complicated patterns. To cope with the problem one has to implement a variety of measures on the IT part and also on the experiment side. Checksum mechanisms have to implemented and deployed everywhere. This will lead to additional operational work and the need for more hardware.

### Introduction

During January and February 2007 we have done a systematic analysis of data corruption cases in the CERN computer center. The major work in the implementation of probes and automatic running schemes were done by Tim Bell, Olof barring and Peter Kelemen from the IT/FIO group. There have been similar problems reported in Fermilab and Desy and information exchange with them was done.
The following paper will provide results from this analysis, a judgment of the situation and a catalogue of measures needed to get the problem under control.
It is also to be seen as a starting point for further discussions with IT, the experiments and the T1 sites.

https://indico.cern.ch/event/13797/contributions/1362288/attachments/115080/163419/Data_integrity_v3.pdf

# Network Corruption

Network router software inadvertently corrupts TCP data and checksum!

XSEDE and Internet2 example from 2013.

Second similar case in 2017 example with FreeSurfer/Fsurf project.

BROCADE

**TECHNICAL SUPPORT BULLETIN**

June 28, 2013

TSB 2013-162-A

SEVERITY: Critical- Service Impact

PRODUCTS AFFECTED:
Brocade NetIron XMR/MLX 100G module (BR-MLX-100Gx2-X and BR-MLX-100Gx1-X).

CORRECTED IN RELEASE:
The fix will be in patch releases of NI 5.3.00eb, 5.4.00d and 5.5.00c and later releases.
This issue is not applicable to software release NI 5.2.00 and previous releases.

BULLETIN OVERVIEW

When transferring data through 100G modules, a portion of the packet may get corrupted. Corruption is typically seen when transferring jumbo frames.

Brocade TSB 2013-162-A

XSEDE

Extreme Science and Engineering Discovery Environment

HOME | ABOUT | USER SERVICES | EDUCATION & OUTREACH | RESOURCES | GATEWAY

News

## XSEDE Network Status

Posted by Bob Garza on 07/25/2013 18:27 UTC

On March 1, 2013 XSEDENet, the network between XSEDE Service Providers, moved to Internet2's Advanced Layer 2 Service (AL2S) national network to take advantage of new features and performance capabilities.

XSEDE was notified recently by Internet2 that an error was discovered on the devices that Internet2 uses on its AL2S network that could possibly lead to data corruption. This error could have affected approximately 0.001% of the data that traversed **each** AL2S device and was undetectable by the standard TCP packet checksum. These errors would have primarily affected data transfers using protocols that did not employ data integrity capabilities (application compression, encryption or checksums). XSEDE users who used secure copy (scp) to transfer files were not affected due to its application layer checksums. Data transfers initiated with the Globus Online web interface also were not affected as Globus Online implemented default checksums in December 2012. Other data transfers including manual gridftp or other protocols without data integrity checking could have been affected by this error.

By July 17, 2013 Internet2, in cooperation with the device vendor, upgraded all the affected devices with a new version of software that corrected the error. XSEDE recommends that users who transferred files using data transfer protocols that do not incorporate data integrity capabilities check the integrity of their file transfers that occurred between March 1, 2013 and July 17, 2013. Please refer to the XSEDE documentation on data integrity and validation of data transfers for details about data integrity checks.

Please submit any questions you may have by sending email to help@xsede.org or by submitting your questions through the XSEDE User Portal @ https://portal.xsede.org/help-desk.

https://www.xsede.org/news/-/news/item/6390

# Software failure

Bug in StashCache data transfer software would occasionally cause silent failure (failed but returned zero).

Internal to the workflow this was detected when input to a stage of the workflow was detected as corrupted and retry invoked. (60k retries and an extra 2 years of cpu hours!)

However, failures in the final staging out of data were not detected because their was no workflow next stage to catch the errors.

The workflow management system, believing workflow was complete, cleaned up, so final data incomplete and all intermediary data lost. Ten CPU*years of computing came to naught.

# Malicious attacks

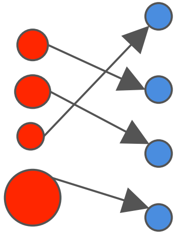
- Script kiddies out for glory.
- Nation-states trying to disrupt/embarrass U.S. science.
- Disgruntled insiders.
- Grad students, post-docs, staff going for that publication with (bogus) phenomenal results.

# Enter application-level checksums

Application-level checksums address these and other issues (e.g. malicious changes).

In use by many data transfer applications: scp, Globus/GridFTP, some parts of HTCondor, etc.

To include all aspects of the application workflow, requires either manual application by a researcher or integration into the application(s).



# Some background

**Hash function** - a mathematical/algorithmic function that takes a set of bits (of any length) and maps them to another set of (hopefully unique) bits of fixed length.

→ primary purpose: detect changes in data

e.g. using a SHA in Python:

```
>>> hashlib.sha256(b"The Answer to the Ultimate Question of Life, the Universe, and Everything is 42").hexdigest()
'8a72856cf94464dd641f0a2620ab604dd7a3f50293784a3a399acf6dc5b651cb'

>>> hashlib.sha256(b"The Answer To the Ultimate Question of Life, the Universe, and Everything is 42").hexdigest()
'a39be9fd272f2569aa95a07134a55f032ecb5c51cef6d66fe4032ec30bf4f1b6'

>>> hashlib.sha256(b"The Answer is 42").hexdigest()
'cbf296e175f02156cd60d6bf93aebd92893e72a0c4c48eadef092d0dc7e28fc1'
```

The fixed length result is the “hash value”, a.k.a. “checksum” or “digest”.

# Our Talk

- Problem Statement: Challenges to Data Integrity
- **Our Approach: Adding integrity support to the popular Pegasus scientific workflow management system**
- Challenges
- Next steps

# SWIP Goals

- Provide assurances that a workflow is not accidentally or maliciously altered during execution.
- Allow for detection of modification to its data or executables at later dates to facilitate reproducibility.
- Integrate cryptographic support for data integrity into the Pegasus WMS.

# Taking Advantage of Pegasus WMS

- Familiar interface to scientific projects (>700k Pegasus workflows from 2013 to 2015).
- Integrity-checking is tedious and error-prone. A WMS system, with its understanding of data ingest and creation is a good place to handle these tasks.
- Manages provenance and metadata, which we can protect.
- Maps abstract workflow to computing infrastructure and with understanding of security needs can choose appropriate infrastructure or even re-configure it.

# Pegasus Workflow Management System

Discover what resources (computation, data, software) are available

Select the appropriate resources based on a architecture, availability of software, performance, reliability, availability of cycles, storage,..

Devise a plan:

- What resources to use
- How to best adapt the workflow to the resources
- What protocols to use to access the data, to schedule jobs
- What data to save

Execute the plan

- In a reliable way
- Keep track of what data was accessed, generated and how

Outside of the WMS functions

- Resource provisioning

# Key Pegasus benefits

- Portability across heterogeneous infrastructure
  - Separation of workflow description and execution
  - Support for campus and leadership class clusters, OSG, XSEDE, academic and commercial clouds
  - Can interact with a number of different storage systems (with different protocols)
- Supports data reuse– useful in collaborations and ensemble workflow runs
- Reliability
  - Recover from failures, retry, workflow-level checkpointing
- Scalability
  - O(million) task, O(TB) data in a workflow
- Restructures workflow for performance
- Web-based monitoring and debugging tools
- Can be included in various user-facing infrastructures
  (Graphical composition tools, Portals, HUBZero)
- Open source, available on github

# Workflow Execution Challenges and Capabilities

**Failures in the execution environment or application**

- Workflow-level checkpointing
- Retries
- Resubmit the workflow onto different resources (pick up where you left off)

**Data storage limitations on execution sites**

- Clean up data as you go along (automatically adds nodes to workflow)

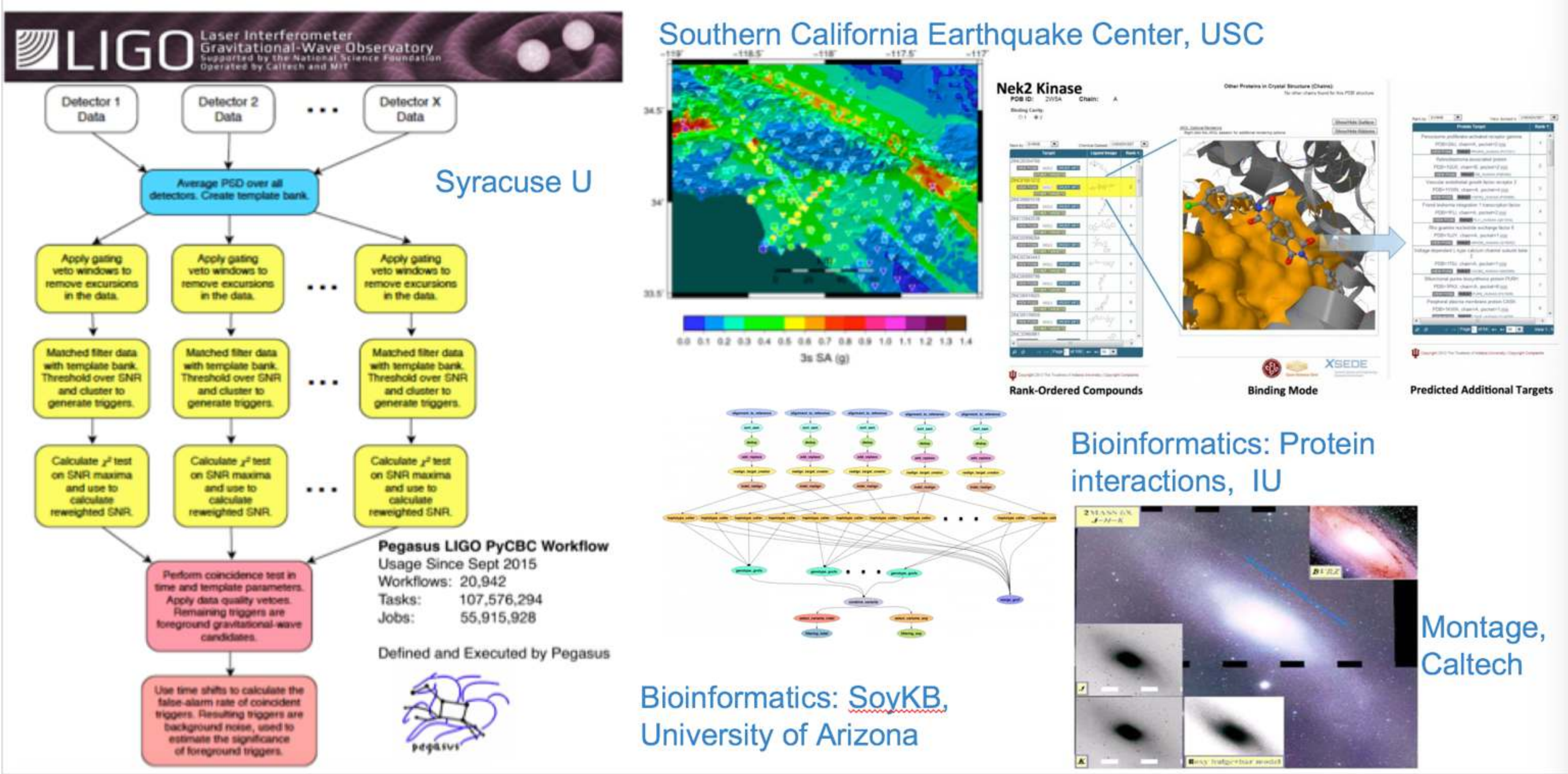
**Performance**

- Small workflow tasks
- Task clustering, pilot jobs
- Data reuse

**Heterogeneous execution architectures**

- Specialized execution engines
- Support for a variety of storage layouts
- Support for most data transfer protocols

# Pegasus, Production quality, In use since 2001

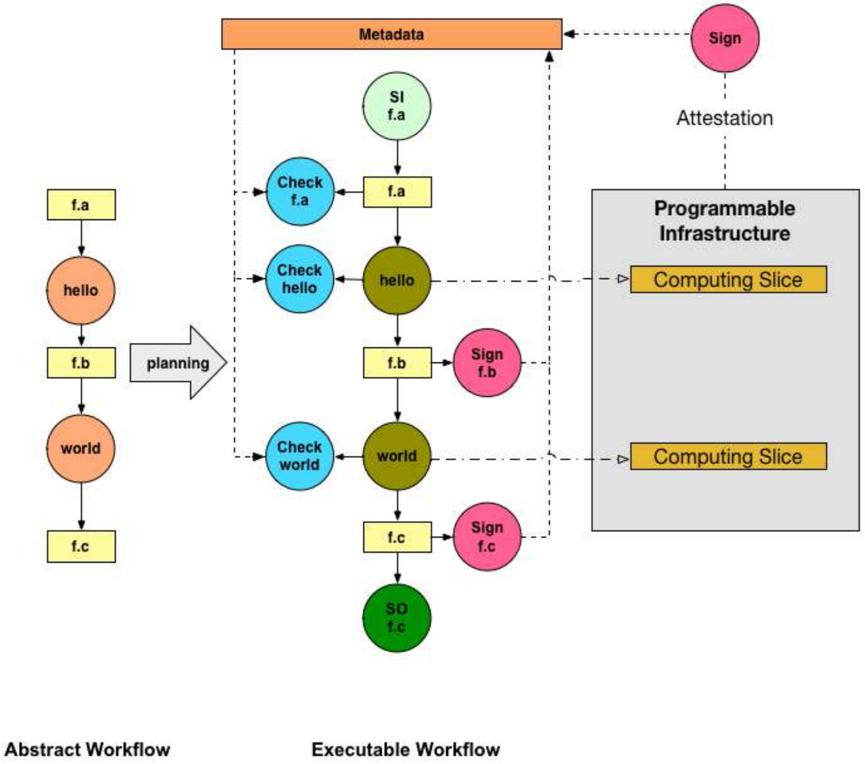


# Our High Level Plan...

Workflow Management Systems (WMS) are great places to tackle data integrity.

They understand what data is created and ingested and do not mind tedious tasks such as generating and checking checksums.

Pegasus WMS is widely used (LIGO, SCEC, SoyKB, Montage, etc.) by the scientific community and is the target of our improvements.

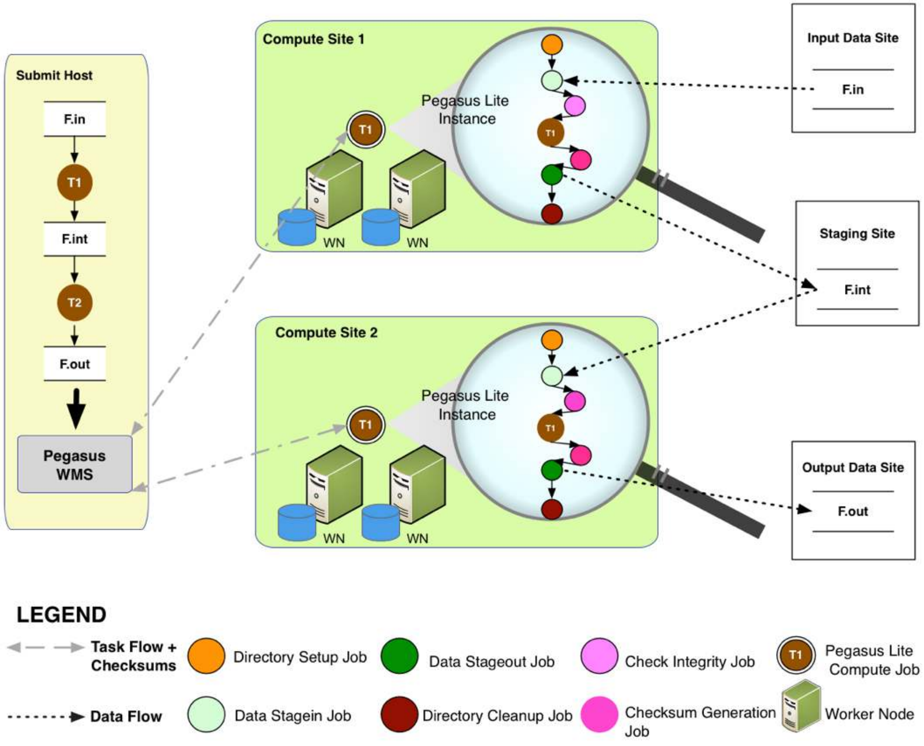


# Automatic Integrity Checking - Current Status

Pegasus will perform integrity checksums on input files before a job starts on the remote node.

- For raw inputs, checksums specified in the input replica catalog along with file locations
- All intermediate and output files checksums are generated and tracked within the system.
- Support for sha256 checksums

Failure is triggered if checksums fail



# Our Talk

- Problem Statement: Challenges to Data Integrity
- Our Approach: Adding integrity support to the popular Pegasus scientific workflow management system
- **Challenges**
- Next steps

# Challenges

Can we do more than know "something changed?"

Balance performance / integrity trade-off?

How do we handle storage without compute capabilities?

Are all errors in all types of data of equal concern?

Long data life: today's cryptographic algorithms will probably not last as long as we need the science data.

E.g. what threats will Quantum computing bring?

When do we hit limits of cryptographic algorithms (collisions)?

→ Prof Steve Myers (Co-PI), IU SICE

# Today’s Limits

We are not modifying operating systems, libraries, and software outside of Pegasus – this limits the strength of the assurances we can provide.

E.g. Modification of system libraries could fool our integrity checks.

As we encounter these limitations we will document how a next generation CI and Hardware stack could address them.

E.g. through the use of trusted computing (Intel Secure Guard Extension, etc.)

# How do you know your integrity protection is working?

<u>Imagine the following:</u>
You finish adding integrity protection to your software. You run a workflow and all goes smoothly.

Was there no integrity problem or did you just fail to detect it?

How do you reliably and repeatedly test integrity protection?

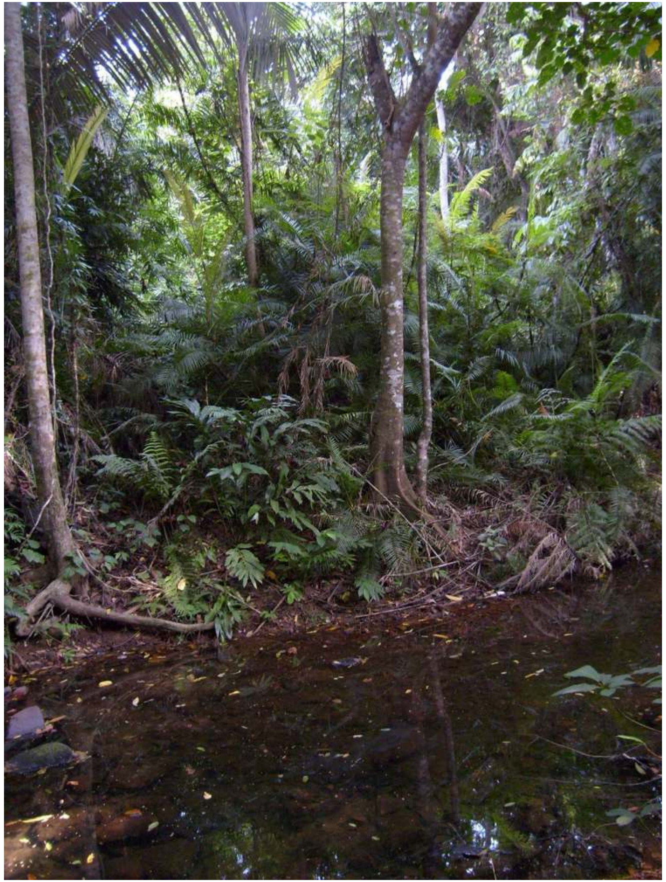
# Enter the Chaos Jungle!

Inspired by Netflix's Chaos Monkey.
https://github.com/Netflix/chaosmonkey

The RENCI ORCA software creates virtual infrastructure. CJ software introduces impairments into data transfers.

Combining the two we get virtual infrastructure that intentionally corrupts data - randomly or predictably?

Now we can test how software runs under bad conditions.



https://commons.wikimedia.org/wiki/File:Tioman_Rainforest.JPG

# Chaos Jungle Primer

Uses Linux eBPF (extended Berkeley Packet Filters) functionality

Introduces a small eBPF program into the kernel attaching to either TC filter or XDP hooks

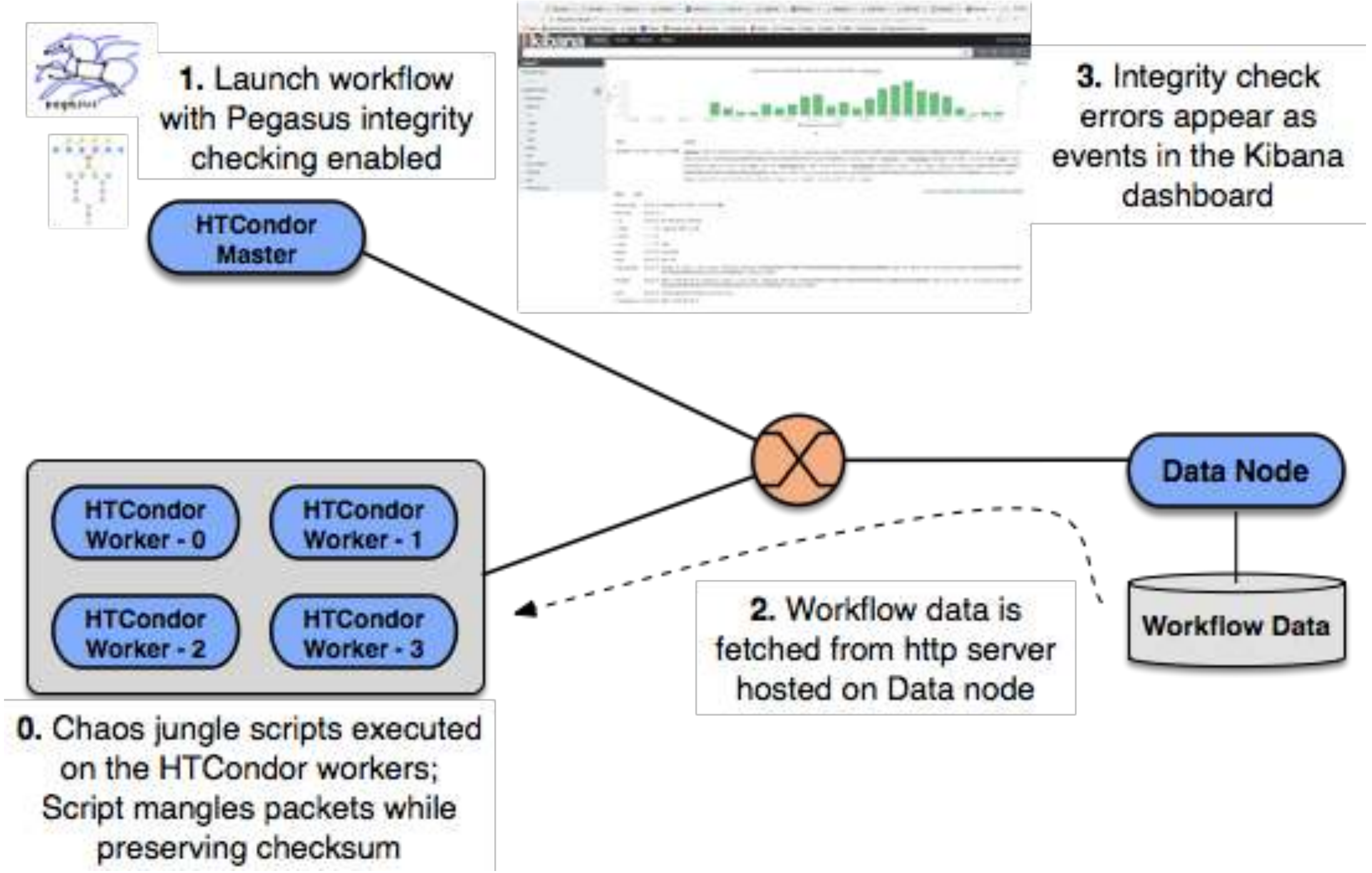
Inspects received packets and modifies some of those that match flow descriptors without affecting the appropriate checksums.

The packets thus look valid on the receiving end, however contain invalid data.

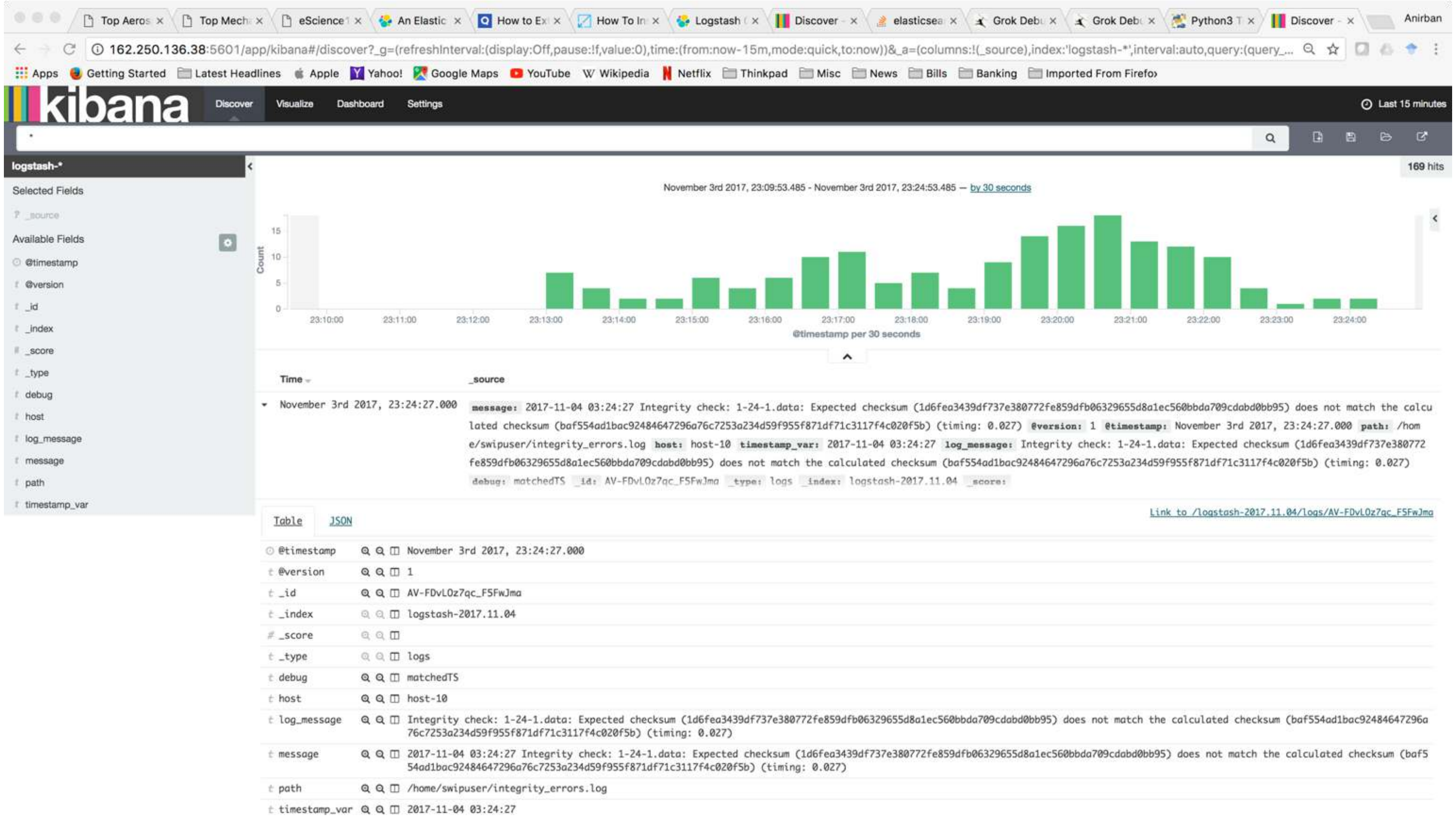
Fast and performant.

https://github.com/RENCI-NRIG/chaos-jungle

# Demonstration Overview



# Demonstration Overview



# Our Talk

- Problem Statement: Challenges to Data Integrity
- Our Approach: Adding integrity support to the popular Pegasus scientific workflow management system
- Challenges
- **Next steps**

# Our three-year plan

<u>Year one:</u> Requirements analysis and prototyping

See: https://github.com/IU-CACR/SWIP/blob/master/SWIP-Community-Use-Cases.pdf

<u>Year two:</u> Iterate with partners to evaluate effectiveness and usability:

LIGO, CyberShake, FreeSurfer, OSG, SPLINTER, Chameleon, NSFCloud

(We are just starting year two.)

<u>Year three:</u> Complete transition to production through release in Pegasus and ORCA.

Will release through existing open-source repositories and licenses.

# Thanks!

SWIP

Scientific Workflow Integrity with Pegasus

https://cacr.iu.edu/projects/swip/

CENTER FOR APPLIED CYBERSECURITY RESEARCH

Pegasus

SICE

renci

We thank the National Science Foundation for funding this work (Grants 1642070, 1642053, 1642090). Views expressed may not necessarily be the views of the NSF. Thanks to Eli Dart for Brocade TSB details.

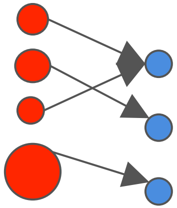
# Practicing what we preach + research

http://download.pegasus.isi.edu/pegasus/4.7.4/sha256sums.txt (current Release version)

```
e58352f89e8325b92d13cac996c029fdc7950b019ea17b9a71a41fadf9ec29a6  pegasus_4.7.4-1+deb7_amd64.deb
94750e8ef2cf381b6b0aaf68ab1412e3763098496b3e3f0b9a74719764ecbdb3  pegasus_4.7.4-1+deb8_amd64.deb
e0a15758815a21c7c1f296842dac079fb14eeb2db624f49f1973b2cd08495baf  pegasus-4.7.4-1.el6.x86_64.rpm
...
26257cfad6eb7e72507a53c49c74f15535ed87475d5fc6ddb9b71b20d8a5afb8  pegasus-worker-4.7.4-x86_rhel_6.tar.gz
```

# Concerns over hash functions

- Collisions, i.e., non-unique hashes
- Computational expense
- “Big data”



# Today's Limits

Scientific Workflow Integrity with Pegasus

We are not modifying operating systems, libraries, and software outside of Pegasus – this limits the strength of the assurances we can provide.

E.g. Modification of system libraries could fool our integrity checks.

As we encounter these limitations we will document how a next generation CI and Hardware stack could address them.

E.g. through the use of trusted computing (Secure Guard Extension, etc.)

# Limitation of TCP Checksum with Big Data

TCP has a 16-bit checksum.

This means 1/65,536 packets will randomly have the same checksum.

So packet corruption is 1/65,536 likely not to be detected by TCP checksum.

In 1999, Vern Paxson found corruption in 1/5000 packets.

V. Paxson, End-to-End Internet Packet Dynamics. IEEE/ACM Transactions on Networking, Vol.7, No.3, pp. 277-292, June 1999 (http://dl.acm.org/citation.cfm?id=312234

Hence:

1/65,536 × 1/5000 =~ 1/300 million

packets will get corrupted and not detected by the TCP checksum.

If we assume 1 kbytes / packet, a 300GB transfer will have one undetected error.