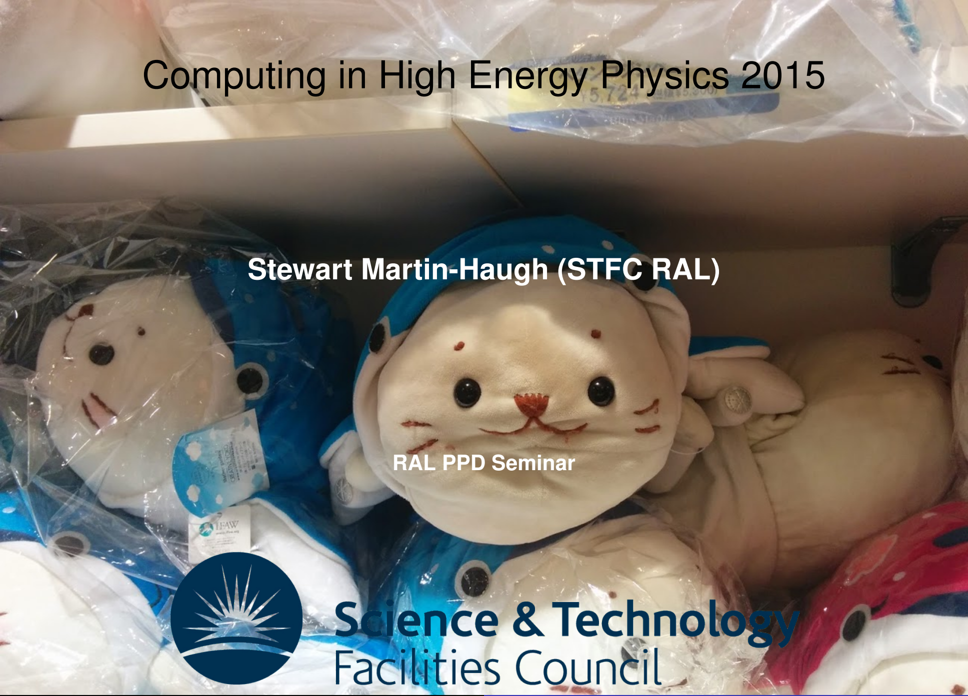
Computing in High Energy Physics 2015



Stewart Martin-Haugh (STFC RAL)

RAL PPD Seminar

Science & Technology Facilities Council

General themes

- CHEP
- Okinawa
- Security
- GPUs
- Multicore
- ROOT
- Offline software
- Online software and computing

CHEP

- 6 tracks
  - Upgrades and parallelism
  - Reconstruction
  - Databases
  - Middleware
  - Computing models
  - Facilities, monitoring
- 11 plenary talks
- 5 industrial talks
- 255 parallel talks
- 248 posters

Okinawa

- Independent Ryukyuan kingdom until 1879
- Distinct culture and language from mainland Japan - strong Chinese influences
- Male and female Shisa "dog-lions": ubiquitous island symbol (and CHEP mascots)
- Birthplace of karate

Okinawa Institute of Science and Technology

- Founded 2007
- Modern campus
- Familiar faces
  - Jonathan Dorfan (President)
  - Ken Peach (Dean of Graduate Studies)
- ... but no HEP group yet (early days)

Security (Sebastien Lopienski, CERN Deputy Computer Security Officer)

- Nice talk informed by
  - High-profile bugs (Heartbleed, goto fail)
  - High-profile hacks (Sony pictures, Github)
- CERN is an interesting case for security
  - Large organisation $\rightarrow$ criminals
  - High-profile science facility $\rightarrow$ hacktivists
  - Hosts critical internet infrastructure $\rightarrow$ governments

Bitcoin mining by a rouge sysadmin

Inspired (issue 18, Feb 2015)
http://www.egi.eu/news-and-media/newsletters/Inspired_Issue_18/cryptocoin.html

"[..] substantial amounts of mining jobs were submitted over the 2013 Christmas holidays before being discovered in early January. The user had attempted to masquerade the mining activities as legitimate production jobs and also tried to hide his traces by planting false evidence of external attacks on the job submission machine.
He failed and was caught."

CERN, “a fun real-world example”

Lab Mouse Security
@InfoSecMouse

Follow

Hacking CERN
Particles and Pr
blog.securitymo
with @DonAndr

RETWEETS 33 FAVORITES 12

1:56 PM - 7 Jul 2014

Hacking CERN - Exploiting python-lz4 for Particles and Profit

TL;DR

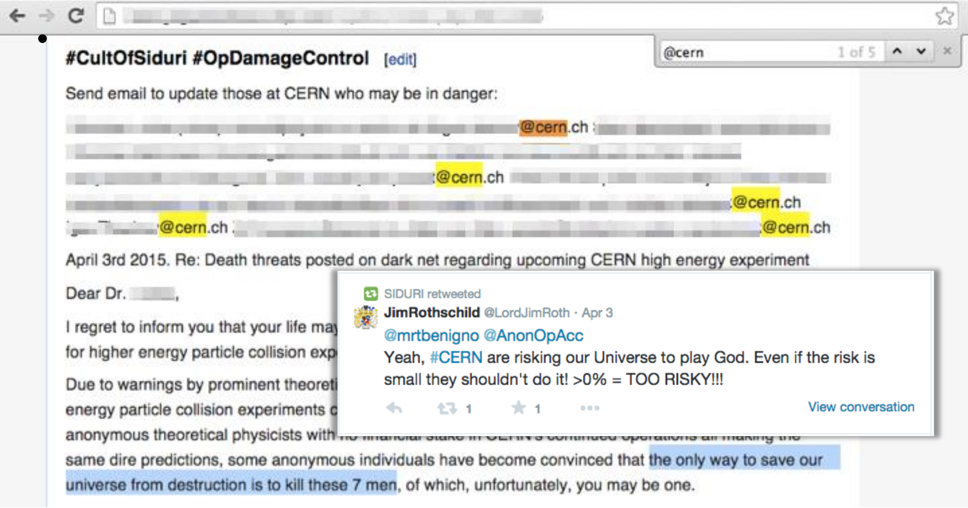
Editor's Note: The TL;DR of this long technical report can be summarized as

- LZ4 was always critically vulnerable whether in Kernel or User-land
- Exploitation is easy regardless of the attack used (16MB or 2+MB)
- PoCs are written for python2.7 on 32bit ARM/x86 (scroll to the end)
- Updating is critical for all consumers of LZ4, not just python-lz4

Additional Note: The author of LZ4 claims that the PoC presented in the blog below was written against some ghostly alternative version of LZ4. For further proof of exploitation, the sample payload generated by the script at the end of this blog post will also crash python-lz4 (versions prior to r119) directly. The CERN software was simply used as a fun real-world example because their package depends on python-lz4. To test, call the Python bindings directly with:

http://blog.securitymouse.com/2014/07/hacking-cern-exploiting-python-lz4-for.html

LHC start-up? Death threats



@cern 1 of 5

#CultOfSiduri #OpDamageControl [edit]

Send email to update those at CERN who may be in danger:

@cern.ch

@cern.ch

@cern.ch

@cern.ch @cern.ch

April 3rd 2015. Re: Death threats posted on dark net regarding upcoming CERN high energy experiment

Dear Dr. ,

I regret to inform you that your life ma
for higher energy particle collision exp

Due to warnings by prominent theoret
energy particle collision experiments c
anonymous theoretical physicists with
same dire predictions, some anonymous individuals have become convinced that the only way to save our universe from destruction is to kill these 7 men, of which, unfortunately, you may be one.

SIDURI retweeted
JimRothschild @LordJimRoth · Apr 3
@mrtbenigno @AnonOpAcc
Yeah, #CERN are risking our Universe to play God. Even if the risk is small they shouldn't do it! >0% = TOO RISKY!!!
1 1
View conversation

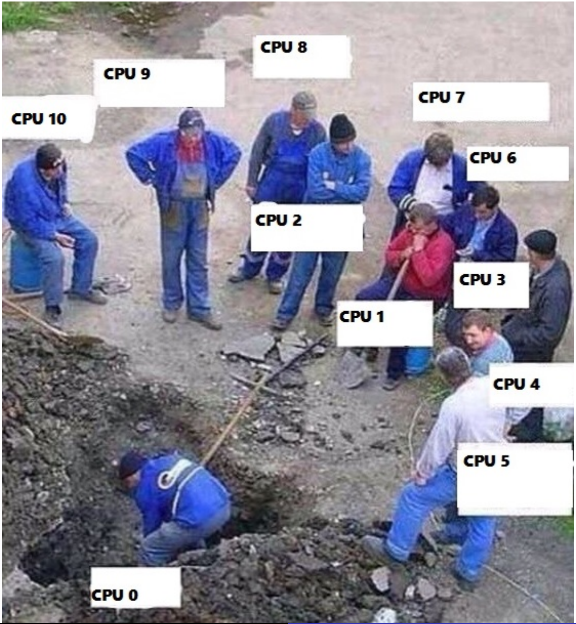
Computer security (Sebastien Lopienski, CERN Deputy Computer Security Officer)

- As usual from security talks, no silver bullet
- Sense that landscape has changed even in last two years after high-profile bugs and hacks

GPUs

- GPUs now mainstream
- Presented by ATLAS, CMS, ALICE, LHCb, T2K...21 talks/posters in all
- Impressive speedups in relevant codes demonstrated (generally at least an order of magnitude)
- ALICE have installed GPUs in HLT farm
  - Not clear to me if they will definitely be used for Run 2
- “Highly specialized facilities (e.g. associative memory, crate-based FPGA trigger) will perform well during Online usage but risk becoming *dark silicon* the rest of the time” (Niko Neufeld)
  - Applies fairly equally to GPUs

Multicore executive summary



Multicore detailed summary

- CMSSW
  - Classify all CMS algorithms as thread-safe or “legacy”
  - Developed clang-based static analyser to determine algorithm thread safety
  - Tests at scale reveal ongoing thread safety concerns - small percentage of jobs crashing
- ATLAS porting algorithms to GaudiHive parallel framework
- No mention of multi-threading from other experiments - not sure of status

ROOT

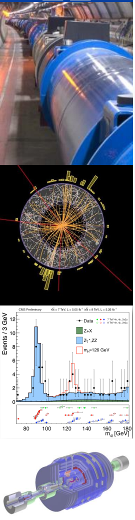
- ROOT 6 migration has turned out to be a huge task (at least within ATLAS)
- ROOT team finishing corners of 6 and looking towards 7
- New logo

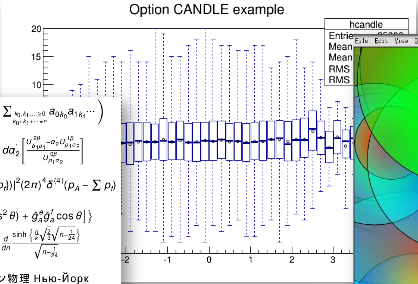
ROOT
An Object-Oriented
Data Analysis Framework

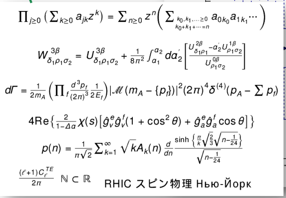
ROOT
Data Analysis Framework

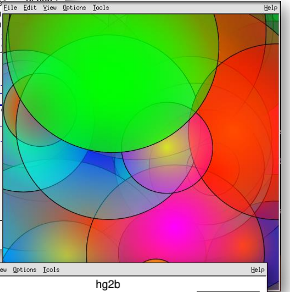
Graphics

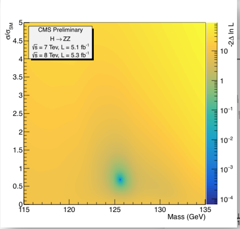
TMathText
Thanks to
Yue Shi Lai

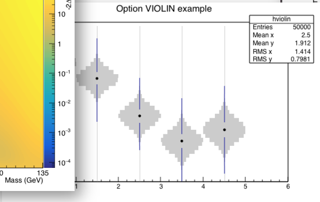


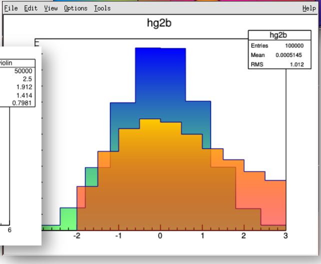




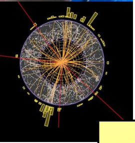


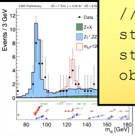


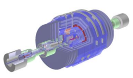




Increase user friendliness







- Many interfaces can be improved in ***C++14,17***
  - Ownership, type safe containers, string options
  - Resulting in improved user productivity
    - Dramatically reduce memory errors, wrong results, etc.
  - Code Self-documentation

```
void OwnOrNot(std::unique_ptr<TWhatever> arg);
 OwnOrNot( & myWhatever ); // Compilation error!
```

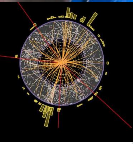
```
// With SetName(const char*)
std::string str; …
std::string sub( str.data()+pos, len );
obj.SetName( sub.c_str() );
```

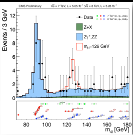
```
// With SetName(std::string_view)
std::string str; …
obj.SetName( {str.data()+pos, len} );
```

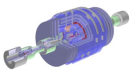
```
// Current
TFile f(name); TH1F h1(…);               f.Write();

// ROOT v7, no implicit shared ownership.
TFile f(name); auto h1 = f.Create<TH1F>(…); f.Write();
```

Interfaces Revolution – v7







- Large existing code base relied upon in production across sciences and continents
  - Must be backward compatible and reuse code base
  - But must evolve the current interfaces

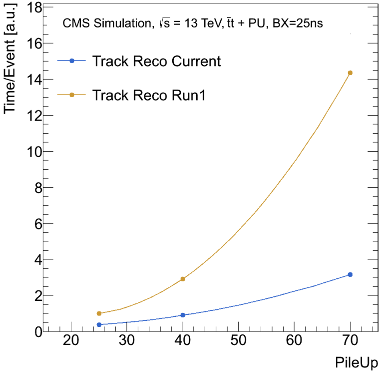
- Gradual introduction of new ***backward incompatible*** interfaces in a 'new' namespace:

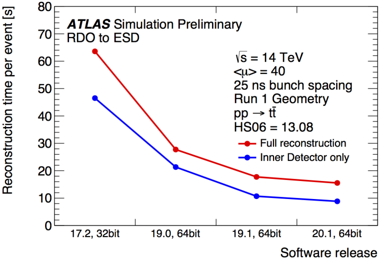
```
  ROOT::T...
aliased with
  ROOT::v7::T...
```

"Things alter for the worse spontaneously,
if they be not altered for the better designedly."
Francis Bacon

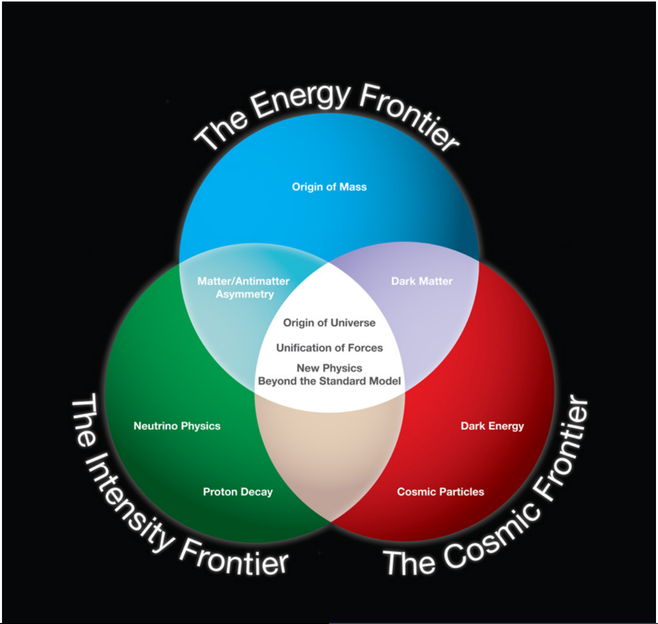
Offline software

- ATLAS and CMS have made similar strides in reconstruction speed for Run 2





Software for the intensity frontier (Craig Group)



Intensity Frontier Experiments

Several categories of IF experiments:

- Quark flavor physics
- Neutrinos
- Charged lepton processes
- New light weakly coupled particles
- Nucleons, nuclei, and atoms

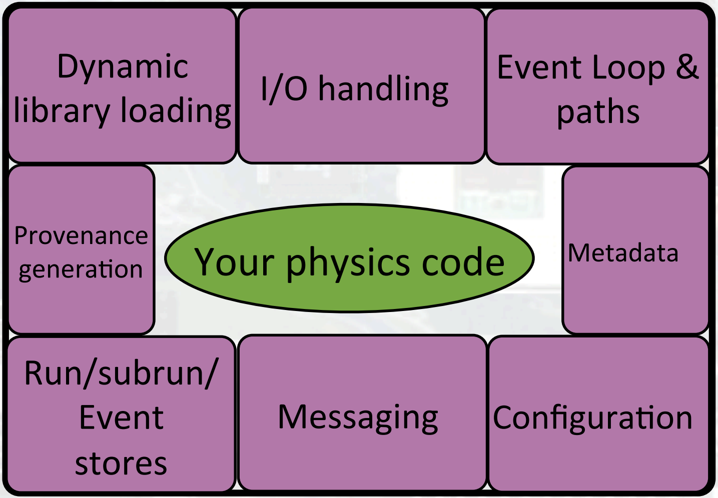
Small Experiments?

- Intensity Frontier experiments are not small!

Airbus-380

- NOvA similar volume to Atlas
- ATLAS – 7,000 tonnes
- NOvA – 14,000 tonnes
- LHC circumference 27 km
- NOvA baseline 810 km

What is a framework?



Code you write **Code you use from the framework**

See slides from yesterday, Liz Sexton Kennedy, track 5:

A Comparative Analysis of Event Processing Frameworks used in HEP

Slide content, A. Lyon – ArtG4 Seminar 2013

Framework Benefits

Allows you to write your physics code without worrying about the infrastructure. Makes it easy to work with others.

→ But not for free – you have to learn how to use the framework!

Some people find such a system constraining:

- Infrastructure is hidden behind the scenes from you
- Your ideas may not be included
- You have to trust a system you didn't write
- You miss out on the fun of writing super-cool complicated C++ code

Some people find such a system liberating:

- You can concentrate on physics code
- Your C++ is pretty easy (you are *using* a complicated system, not *writing* it)
- You get to miss out having to maintain the complicated C++ code
- You can use code from others and share yours with others
- You can get services for free (e.g. data handling)

Slide content, A. Lyon – ArtG4 Seminar 2013

Why not write your own framework?

"Small" experiments may not have…

- The expertise
  - Writing large C++ systems is hard (need low dependences, efficient generic programming, follow software engineering best practices)
- The time
  - With lots of milestones and reviews, there's no time to devote to correctly writing such a large system
- The energy
  - We just wanna make plots! Not write infrastructure code

Slide content, A. Lyon – ArtG4 Seminar 2013

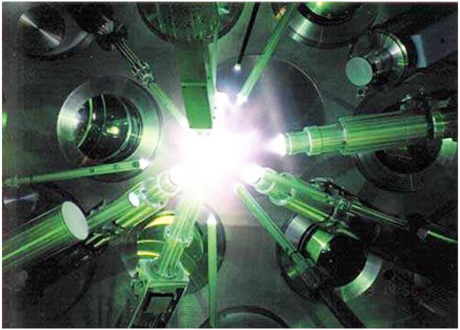
The art Framework

- art is a framework forked from CMSSW and tailored for neutrino/muon experiments.
- Experiments use art as-is as an external. art **is not** modified by the individual experiments. Used like ROOT, G4, boost...
- Used by Mu2e, Muon g-2, NOvA, MicroBooNE, LAriAT, Darkside-50, and DUNE prototype efforts.
  - Easier for individuals to work on multiple experiments -- this is a common practice for neutrino experiments
  - Experiments can share solutions to common problems
  - Common training (classes, workshops, workbook…)
- Seamless integration to data-handling tools (I/O) is an important aspect of art.
- New features and direction decided among stakeholders by consensus.
- Main support forum is centralized, but experts from experiments help answer questions.

Online software and computing (Niko Neufeld)

- Commonality between different experiments
- Large facilities outside HEP have similar requirements

NIF

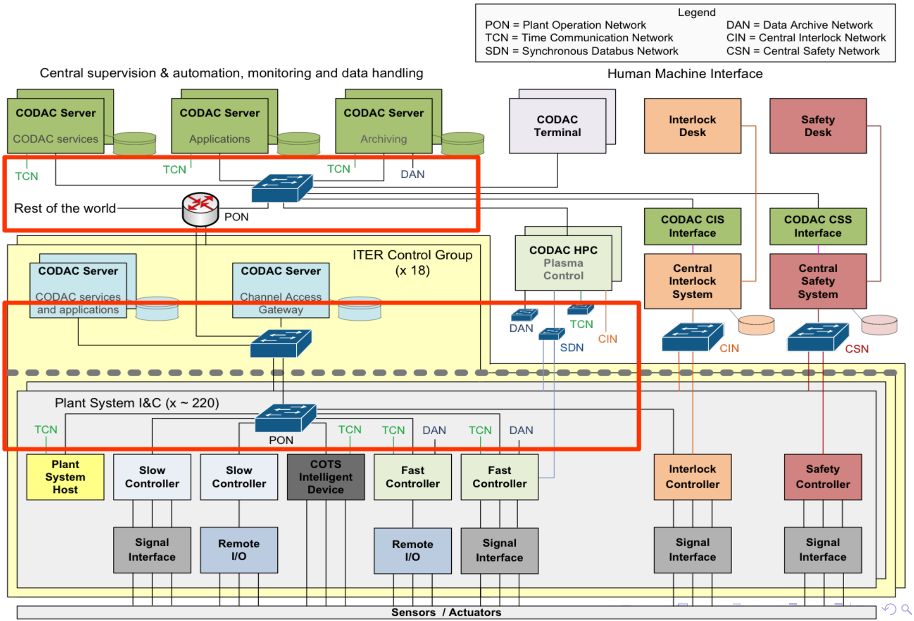


source: http://www.newswise.com/images/uploads/2011/04/29

- Facility as complex as the LHC
- 4 million components

- Control system based on 500 physical PC servers, about 3000 VMs
- More than 2500 servers total, 750 TB storage
- Continuous DAQ from a very large number of front-end devices, sensors, ... [2], in total 4000000 individual components (not all on the networked)
- Migrated from Solaris, VxWorks, Ada...
- to Linux and Windows, Java and CORBA

ITER control - CODAC



CODAC in numbers

# computers	1000
# signals	100000
# process variables	1000000
data rate for acquistion	up to 50 GB/s
archival	90 - 2200 TB / day

- As a DAQ as large as LHCb during Run1
- Mostly commercial components
- Ethernet even for real-time systems and timing distribution
- Many bursty streams
- Can be seen as a large “Internet of Things”

Evolution of LHC DAQ

- All systems evolve and show remarkable capability of smooth change
- Particular nice example: CMS (Myrinet $\rightarrow$IB, file-based HLT)
- Convergence to single-stage HLT (ATLAS, CMS, LHCb)
- Combination of synchronous and asynchronous HLT (LHCb already at the end of Run1, ALICE from Run3)
- All this while preserving a large part of the enormous amount of work invested in these systems (and even some of the hardware)
- All of them are excellent examples of good, architecture-driven design (, OK I am not unbiased here :-))

Conclusion

- LHC experiments prepared for Run 2 (similarly for non-LHC experiments)
- Commoditisation of specialised LHC computing and software - use of more off-the-shelf components
- Same ideas often pursued/arrived at separately by different collaborations or even those outside HEP
- End of Moore's Law nigh, no consensus solution
  - Multi-threading existing serial code difficult
  - GPUs popular solution but not yet fully mainstream