Linux: the Engineer's Tool Box

Computer Tech 2018

Tom Browder [tom.browder@gmail.com]

Northwest Florida Linux User Group [nwflug.org]

2018-01-27

AUDIENCE EXPECTATIONS

What do you expect from this presentation?

OBJECTIVE

My purpose is to give you an introduction to the utility of a Linux computer for scientists and other technical persons.

I hope the parents and grandparents of children interested in a technical career can benefit from the information and pass that on.

Background

- Retired USAF fighter pilot, retired engineer with ManTech Corporation
- Programmed computers since 1961, had my own programmable computer(s) since 1974
- Used Unix and GNU/Linux and FOSS professionally for over 23 years
  - Use them daily in my personal projects as well as my work on FOSS projects
- I'm curious about lots of things, I'm passionate about computer programming

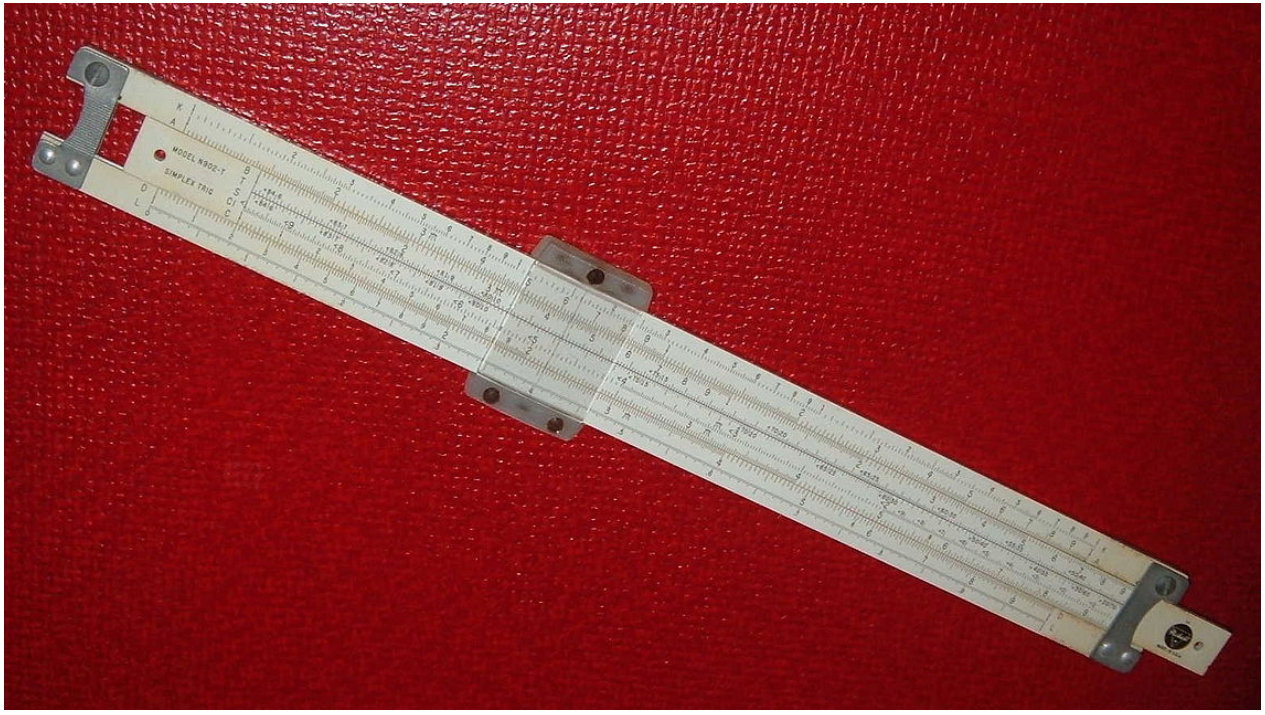
Engineer's Tools

In my college days (early 1960's) we used:

- slide rule
- programmable calculator
- mathematical tables (CRC Tables)
- desk calculator
- *mainframe digital computer
- *analog computer

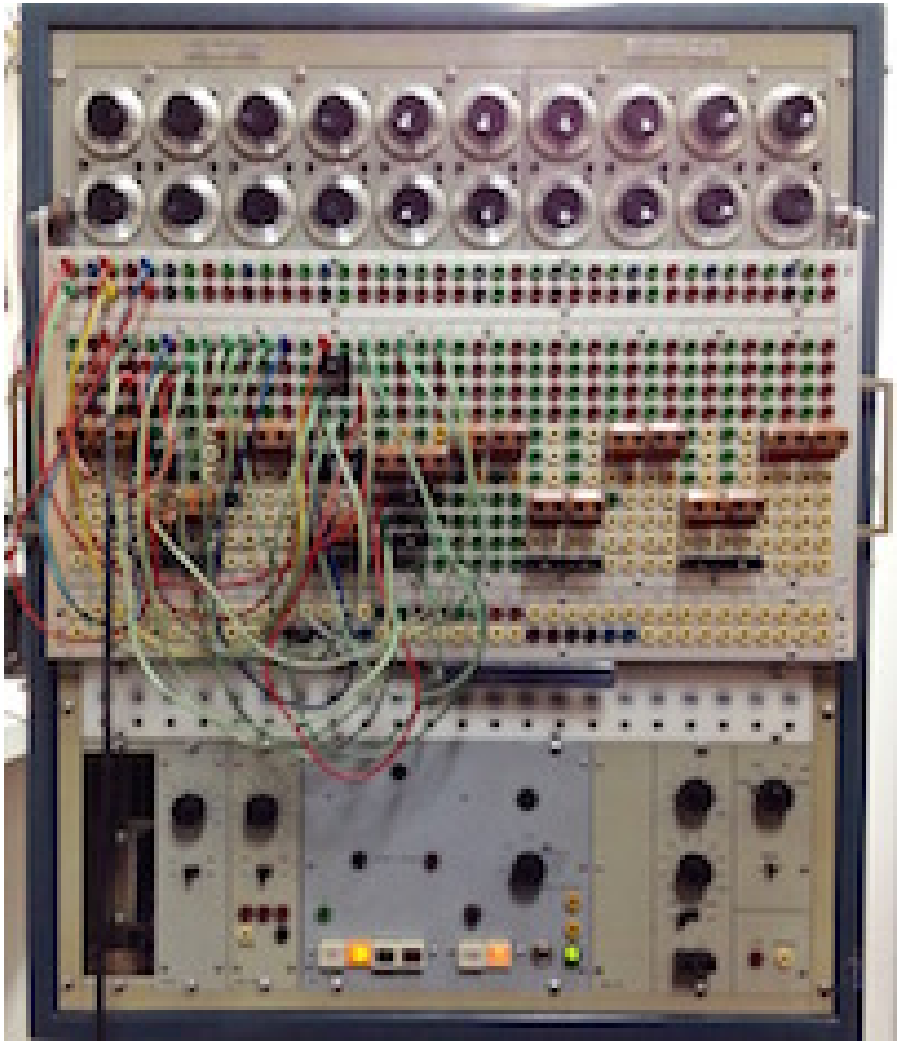
* shared use

Slide rule similar to the one I used



Small analog computer

First EE course in analog computers:



Large analog computer

Advanced EE course in analog computers:

Computer room from the mid-1980's

On the left is the late Mike Muuss, inventor of **BRL-CAD** and the well-known freeware utility *ping*.

Engineer's Tools

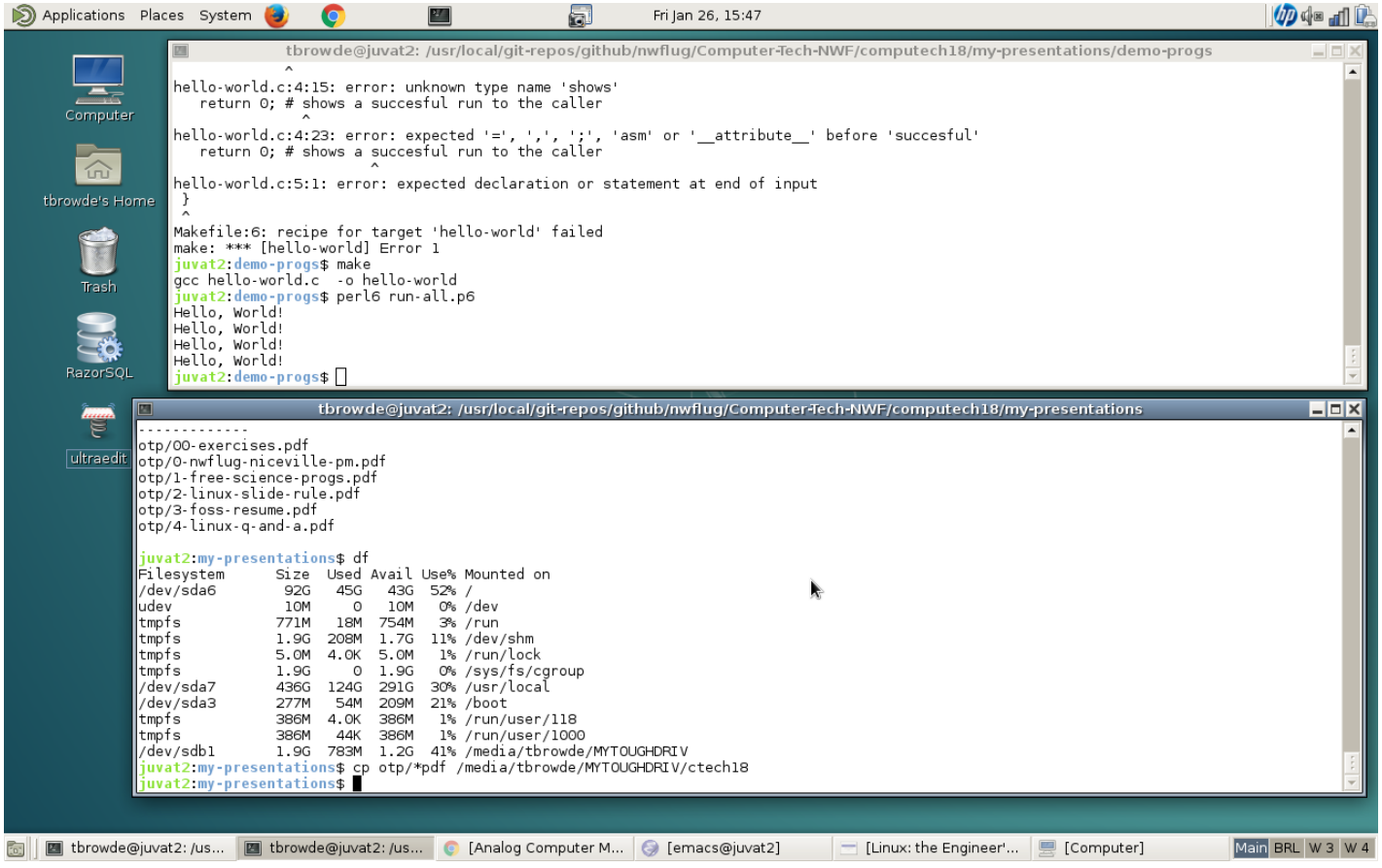
Current day engineers (and scientists, economists, artists, mathematicians) use

- personal computer
- *cloud computers

* shared use

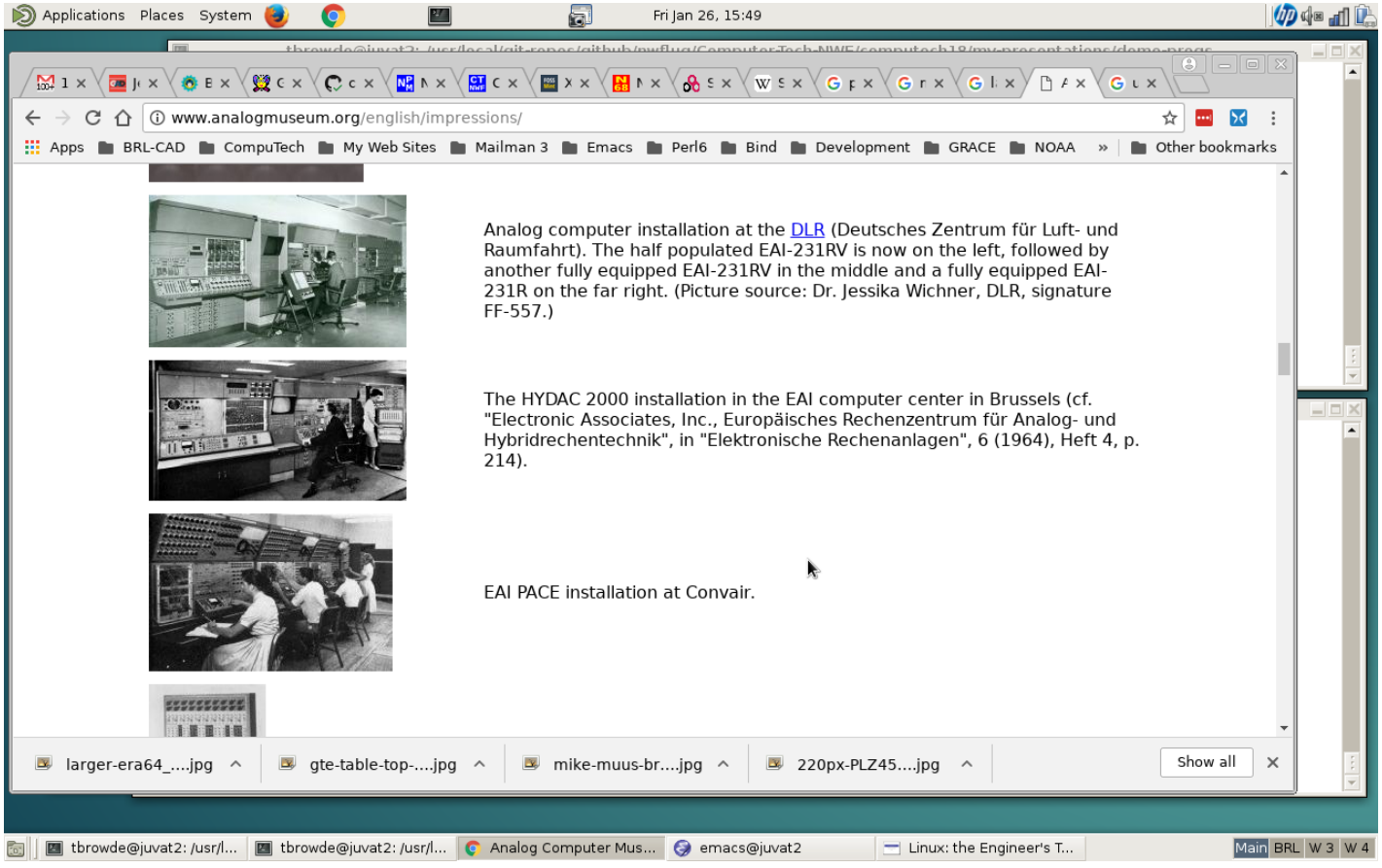
My Debian 8 Linux Desktop

Empty while working at the command line. . .



My Debian 8 Linux Desktop

Cluttered while working with windowed programs. . .



Why Linux?

- Tool box atmosphere
- CLI more usable
- Multidisciplinary
- Kitbashing (chaining tools)
- More control over your system

Windows vs. Linux

- Windows:
  - Is everywhere
  - It's primarily a GUI operation
  - Expensive
  - **Hides internals**
  - **Non-case-sensitive file system**
- Linux:
  - Is pretty much the opposite of Windows in most respects
  - Is unparalleled for the scientist or engineers: a powerful, "programmable calculator"
  - **Is free**
  - **Ugrades are done ONLY if you want to!**

Note that in Linux we say *directories*, **not** *folders*!

Kitbashing

The Urban Disctionary define *kitbashing* as *The practice of modifying a model (not limited to toy action figures) to achieve some result other than that intended by the manufacturer.*

I use it similarly in that, given a particular task and resulting work flow, a person may have to use a model (program) that is reswtricted in some way, and envelope it to get the desired results.

Kitbashing Example

Some commercial programs I've seen require you to take the following steps:

- tediously hand-enter data for individual *cases*
- run the *program which outputs results in some rigid text format (may be difficult to *parse*)
- post-process the output into desired form

That manually-intensive work becomes almost impossible if the set of *cases* gets too large (or at least the time and costs get too expensive for both parties).

* The saddest cases are the programs which output data in their own proprietary format which only their post-processor can use! Keep as far away as possible!

Kitbashing Example

With *Linux* one can usually find a way to at least partially automate the task anyway. One **tool** to help do that is **Expect** which is under some versions of Windows, including the **Cygwin** or other work-around environments running under Windows.

See its website here:

- <http://expect.sourceforge.net>

Kitbashing Example

Let's say we are to evaluate a new kinetic energy round against a military target, such as this:

Kitbashing Example

Use *pseudocode* to define your work flow in program runs, e.g.:

Collecting data using **BRL-CAD** as the ray-tracer:

```
for every velocity
  for every impact angle
    for every aspect angle
       run 'rt' against the target
       collect raw data
    end for
  end for
end for
```

Kitbashing Example

Analyzing data

```
for every velocity
  for every impact angle
    for every aspect angle
       arrange the data for input to a graphing program
       arrange the data for a table
    end for
  end for
end for
```

Kitbashing Example

After you get work and data flows established in pseudocode, you can then script repetitive cases with, my choice, **Rakudo Perl 6**:

```
#!/usr/bin/env perl6
my @vels = <1000 2000>;
my @angs = <30 45>;
my @azs = <0 180>;
for @vels -> $v {
    for @angs -> $ang {
        for @azs -> $az {
            run "rt", "$v $az $ang";
        }
    }
}
```

SUMMARY

- We have taken a brief look at using Linux as an engineer's tool box.
- If we have time, we can use a volunteer to look around and maybe do some simple exercises on a Linux laptop running Linux Mint 18.2.

Finally

- **Please make sure you're on the attendance roster.**
- **Feel free to contact me at any time regarding this session or any other of my sessions (please mention CTech '18 in the subject):**
  - **Tom Browder**
  - **tom.browder@gmail.com**