The DTraceToolkit Project Has Ended

15 May 2015

I originally posted this at http://bdgregg.blogspot.com/2015/05/the-dtracetoolkit-project-has-ended.html.

Ten years ago on this day I created the DTraceToolkit, and it's time to call the project ended. Its scripts live on in different operating systems: OS X, FreeBSD, Oracle Solaris 11, and other Solaris derivatives; as packages for OmniOS and SmartOS; and integrated into other tools. Thanks to everyone who helped make it a success.

I documented its origin in the History file:

```
$ more DTraceToolkit-0.99/Docs/History
-------------------------------------------------------------------------------
20-Apr-2005     Brendan Gregg   Idea
        For a while I had thought that a DTrace toolkit would be a nice
        idea, but on this day it became clear. I was explaining DTrace to
        an SSE from Sun (Canberra, Australia), who had a need for using
        DTrace but didn't have the time to sit down and write all the
        tools he was after. It simply made sense to have a DTrace toolkit
        that people could download or carry around a copy to use. Some
        people would write DTrace tools, others would use the toolkit.
-------------------------------------------------------------------------------
15-May-2005     Brendan Gregg   Version 0.30
        I had discussed the idea of a DTrace toolkit with the Sun PAE guys in
        Adelaide, Australia. It was making more sense now. It would be much
        like the SE Toolkit, not just due to the large number of sample
        scripts provided, but also due to the role it would play: few people
        wrote SE Toolkit programs, more people used it as a toolkit. While
        we would like a majority of Solaris users to write DTrace scripts,
        the reality is that many would want to use a prewritten toolkit.
        Today I created the toolkit as version 0.30, with 11 main directories,
        a dozen scripts, man pages and a structure for documentation.
...
```

Back in 2005, the DTraceToolkit was a collection of robust performance tools for a single OS and kernel, providing advanced performance insight beyond the typical Unix toolset. I've had countless emails from sysadmins and developers who have used it to solve performance issues in production, and wanted to say thanks. I've appreciated the kind words!

Today, in 2015, the 230-script DTraceToolkit is more like a large collection of ancient kernel patches that, when they do work, often do so by sheer luck. I didn't know in 2005 that DTrace would appear on other operating systems, and that each kernel would change as much as it did. In particular there were numerous changes to the DTrace syscall provider, which caused the scripts to be more tied to kernel versions than expected.

To put this all in today's terms: the DTraceToolkit became like a set of 230 *amazing* Linux 2.6.11 patches that people want working on Linux 3.2, 3.13, and 4.0, *and* FreeBSD 10.0, 11.0, *and* Oracle Solaris 11! Such a feat isn't impossible, in the strictest sense of the word, but it is impractical.

In 2013 I saw how to fix this, while keeping the DTraceToolkit as a central collection of scripts. The trick was realizing that there were two audiences with different requirements, who could be served by having two different collections of tools:

- **A toolkit** of working and maintained tools for everyone to use. It should be easy to learn, providing simple Unix-like tools with man pages and example files. It should also provide the fewest tools possible (fewer than twenty), to make it easier to learn, browse, and maintain.
- **A toolshed** of in-development or unmaintained tools for performance engineers to browse. This would be a large library of hundreds of scripts, most of which won't work on any given kernel. These serve as ideas, suggestions, and starting points for performance analysis, and can be fixed when needed. (In a way, the DTrace book serves as this; its scripts are on dtracebook.com and github.)

I planned to do this split, and started explaining it in my 2013 post on DTraceToolkit 0.xx Mistakes. It would be a lot of work. I was primarily interested in helping the toolkit users, but I had another minor motivation: to give a DTraceToolkit talk at the USENIX/LISA conference – a dream I'd had for years. Despite giving many talks, and including the DTraceToolkit in some of them, I'd never given a canonical DTraceToolkit talk.

However, it was not to be. In March 2014 I stopped working on Solaris or any of its derivatives, and accepted a job to work primarily on Linux performance. While the DTraceToolkit was always a spare time project, I have to admit that it's no longer a priority, and hasn't been for years. There are new and exciting things in tech to work on (including Linux eBPF), and which are more related to my day job and career going forward.

I still use some DTrace ... on FreeBSD, which is also in use at my new job. And last year I released a new set of (toolshed-like) tools for FreeBSD: the DTrace-tools collection.

I expected my new job to be the most challenging of my career, and it has been. Early on I desperately missed the DTraceToolkit while on Linux, as well as my other DTrace tools. But I've been writing new ones, out of necessity, based on Linux ftrace, perf_events, SystemTap, and eBPF. I'm making progress, script by script, bringing the observability I need to Linux, and sharing these scripts online.

I gave my LISA talk after all, in 2014, but it was titled Linux Performance Analysis: New Tools and Old Secrets. This was about my Linux ftrace and perf_events-based tools: perf-tools, which are inspired by my own DTraceToolkit. It includes multi-tools like funccount and kprobe, which, for me, make a giant difference. I can rapidly explore kernel behavior again.

A number of the DTraceToolkit scripts live on in different OSes, and I'm glad for that to happen: they have a life of their own. But I can't recommend anyone continue the DTraceToolkit project. If you want to understand more background as to why, then my mistakes blog post should be a good start.

I'm proud of what I accomplished with the DTraceToolkit, the DTrace book, and with Sun Solaris in the field of performance. Thank you Sun – you were awesome back in the day.

I'll be moving the DTraceToolkit into my Crypt, which has my other retired Solaris software. It's not easy to do this, but I think it's better to communicate bad news than none at all.

```
dtrace -n 'END { printf("The %stoolkit has %sED\n", probeprov, probename); }'
```

Probably the best outcome of my DTraceToolkit work was my DTrace book with Jim Mauro. I really think of it as the DTraceToolkit version 2.0.

PS. This is my last post to this blog, which mostly existed for DTraceToolkit updates. My new blog is here.