SystemTap

August 2017 OLUG

by Aaron Grothe

What is SystemTap?

SystemTap starts with a project named Dtrace

Dtrace is a dynamic tracing framework made by Sun Microsystems and designed to be used with their Solaris Operating System in 2005

Dtrace was ported to Linux in 2011

What is cool about DTrace?

- DTrace can be turned on for programs without source
- DTrace can be doesn't require recompiling the program
- DTrace doesn't impose a performance penalty when it isn't running
- DTrace doesn't require re-installing a program

So why isn't this a talk about Dtrace?

- DTrace is licensed under the CDDL license
  - This is the same license that ZFS is licensed under
  - Dtrace is available as part of Oracle Enterprise Linux
- Quite simply you can take the source for DTrace and add it to your own Linux system but you can't redistribute that version as the CDDL and GPL licenses aren't compatible - differing opinions on this
- Given the licensing uncertainty the SystemTap project was started to create a system that would be similar to Dtrace but licensed under the GPL

Installing SystemTap

- SystemTap is available in almost all major distributions
- For Debian the following should work
  - # apt-get install systemtap
  - # apt-get install linux-image-`uname -r`-dbg
  - # apt-get install linux-headers-`uname -r`
- There are pages on the Redhat/Fedora/Ubuntu sites on how to do the installs on those OSes
- Linux Mint doesn't offer -dbg so you have to build your own packages
- Largely breaks down to you need the debug for the kernel, headers and systemtap
- You also need gcc if you use the C extensions

Using SystemTap

- There are two or three groups that will be created when you install SystemTap
- StapUsr - group to run a compiled module
- StapDev - group to be able to build a compiled module
- StapSys - group if you're doing client/server install

Simplest to just add your relevant users to stapusr and stapdev groups as that gives needed permissions to build/run modules

How Does SystemTap work? (BTS)

- Create a .stp file in systemtap language
- Is converted to "C" by Stap process
- Compiles into a kernel module .ko
- Staprun Inserts kernel module into running kernel
- Staprun Runs Kernel module (until Control-C, or programmatic exit)
- Removes it from kernel when done

How it looks to the user

- Create Stap Script
- Stap script.stp
- Output from script until you stop or it exits

How does it compare to other tools?

There are a lot of other tools that can be used for profiling

Perf - performance monitoring tool
Lttng - linux tracing tng - kernel level monitoring
/proc - the data is in there
Strace - process trees
Many others

SystemTap seems to be a pretty good mix of functionality with a decent implementation language

Time for Hello World

- Almost everybody starts with Hello World and so will we
- This isn't a bad idea because it also provides you a simple/known test that your systemtap install is working correctly

Time for Hello World

Hello.stp

```
Probe begin
{
    print (“hello world\n”)
    exit () // needed as program won’t terminate without
}
```

Time for Hello World

stap hello.stp

Output should be “hello world”

We should do a Demo

Startup firefox
Go to the olug website
Firefox dies, ok lets try it again
Firefox dies again
What is going on
We'll fire up sigkill.stp - tells us how it is dying
    Stap sigkill.stp
Now we need to find out what is doing sigkill
    Stap pstracing.stp
Ok now we now we'll fix that and see what happens

We should do a Demo (cont'd)

Ok.
Now we now the process is receiving a sigkill by uid:1000 (grothe)
We also now what process is doing the killing cron
Time to take a look at the crontab
We'll comment that out for the time being
Everything is hunky dory now :-)

Demo Notes

So that is a very small example

It shows a bit of the power that is available with SystemTap. If you install the systemtap-docs you'll have access to some examples under /usr/share/doc/systemtap-doc or someplace similar
Broken down into
    Memory, io, network, games, process, profiling and more
Many more examples are available at SystemTap wiki

Isn't SystemTap Dangerous?

- SystemTap tries to be friendly to the system
- It has internal limits that hold back processing time so hopefully it doesn't monopolize system
- In the past there have been priv escalation bugs in SystemTap since you are running with some privs
- Guru mode which we'll talk about later could be quite dangerous

Examples - Quick Look

Time to look at some random examples SystemTap scripts

We'll take a quick look at a couple in the io folder

Pulled these examples from multiple sources

Guru Mode

- We are going to disable "kill -9" on the system via a SystemTap script
- One of the first laws of Unix was that every process must accept and die when it gets a "-9"
- Guru mode allows you to "break" more rules. One of the top goals in SystemTap development is to add more features and hopefully eventually remove guru mode all together
- Time to give it a spin - immortal.stp - away
- This example comes from the SystemTap Wiki / War Stories page

Guru Example (cont'd)

- This example was pulled from the War Stories page at the sourceware.org/systemtap page - immortal example
- This program shows how to modify the sigkill requests on the system so they are dropped

Pitfalls - one of these will bite you

- Make sure you have Linux Headers and -dbg installed or your SystemTap may fail in “interesting” ways
- Make sure you read the documentation for the examples as some of them expect parameters and will fail if you don’t provide them
- Don’t install the systemtap-client/systemtap-server stuff unless you need it, once you install them it will try and compile it for client/server and you have to override it

Advanced Capabilities

- SystemTap supports a client/server environment where you can compile scripts and then deploy them to other systems
- Python/Perl/Ruby libraries to make it easier to parse results
- There is a GUI available for SystemTap, seems you have to compile it from source - hasn't been updated in a few years :-(
- GnuPlot used by some people to create pretty pictures
- Systemtap with Guru mode supports "embedded-c" c code copied directly into kernel :-)/:-(

SystemTap enhancing your Programs

- Dtrace has the concept of markers that you can insert into your programs this makes it easier for regular users to use them
- MySQL can be built with the markers - has roughly 200 of them for operations such as open/close/read/write, etc
- If you add them to your code you can get a lot of additional information from your customer as to why something isn't working in the field

SystemTap enhancing your Programs

- PostgreSQL has these already
- We'll take a look at a quick summary of them
- https://www.postgresql.org/docs/devel/static/dynamic-trace.html
- There are not on by default, but are interesting

Other Tools

- FreeBSD/NetBSD and MacOS have dtrace available
- Ubuntu could theoretically have dtrace made available on it since they have already bitten the bullet and have made the ZFS filesystem available
- Oracle Linux has Dtrace available
- Solaris has Dtrace of course
- There was a project to put SystemTap on the BSDs it looks to be dead at the moment :-(

SystemTap Website

- http://sourceware.org/systemtap

Summary

- SystemTap is aninteresting project. It is available on most Linux versions - you may have to do some work for it though
- It has been used by groups such as Java/PHP/LusterFS and MySQL people to help them figure out performance problems
- If you have a custom app you run it can teach you a whole lot about where your system is spending its time
- It can also let you change the behaviour of a system when you don't have access to the source

Summary

- SystemTap has been around for over 10 years so it is pretty well understood
- There are a very nice beginner guide/tutorial for this available in systemtap-doc and the sourceware homepage

Links

- sourceware.org/systemtap - website for systemtap, latest information and the wiki is really nice
- Access.redhat.com - RedHat's documentation website - search for SystemTap - has some very nice little demos available
- slideshare.net/posullivan/monitoring-mysql-with-dtraces ystemtap - example of how to systemtap enable your own programs
- /usr/share/doc/systemtap/examples/index.html - local documentation copy, most of the examples should work on your system

Q & A

Questions?