TEXAS ADVANCED COMPUTING CENTER

WWW.TACC.UTEXAS.EDU

TEXAS
The University of Texas at Austin

10 years of XALT: A Retrospective

Robert McLay

August 17, 2023

XALT: Outline

XALT

- History: Lariat and ALTD
- XALT 1 funded by NSF
- XALT 2
- Sharing Memory and Namespace with every program run on the system
- Lessons learn from running an Open Source Project
- Conclusions

History

- My Boss, John Cazes, asked if I could track what MPI executable ran on our systemt
- I added code around our ibrun submission tool (AKA our mpirun/srun)
- Could do some tracking but no database
- Lariat was born (was LASSO but name conflict)

XALT History

- Mark Fahey contacted us at TACC for a joint proposal
- With an amazing amount of help from my colleague Doug James
- Mark and I joined forces to submit a successful NSF proposal in 2013.
- XALT combined the best parts of both projects (ALTD and Lariat)
- The name XALT comes from eXtended ALTd (suggested by Doug James)
- Originally UTWrangler but name conflict again.

XALT I

- It only tracked MPI executions
- It did have a database
- It tracked Linking of executable
- Used UUID (universal unique id) to connect links to execution
- Rather than making calls to database like ALTD did
- It mainly used Python to colllect data (a bad idea)

XALT 2

- My friend and colleague, Bill Barth, found this ELF trick.
- One could modify the execution of non-static execution.
- By using LD_PRELOAD and a shared library.
- There are similar ways under macOS but ...

How XALT 2 works

```
#include <stdio.h>
void myinit(int argc, char **argv)
{ printf("This is run before main()\n"); }
void myfini()
{ printf("This is run after main()\n"); }

__attribute__((section(".init_array"))) __typeof__(myinit) *__init = myinit;
__attribute__((section(".fini_array"))) __typeof__(myfini) *__fini = myfini;
```

- my_docs/22/xalt_monthly_mtg_2022_03_17/code/bad_memory/ex1

How XALT works (II)

```
% cat try.c

#include <stdio.h>
int main()
{
  printf("Hello World!\n");
  return 0;
}
```

How XALT works (III)

```
$ ./try

Hello World!

$ LD_PRELOAD=./libxalt.so  ./try
This is run before main()
Hello World!
This is run after main()
```

- my_docs/22/xalt_monthly_mtg_2022_03_17/code/bad_memory/ex1

Hijacking the linker

- XALT uses Lmod to put its ld early in the path
- This doesn't work with llvm compilers
- Sites have to move llvm's ld to ld.x and place XALT's ld in the right place

Implications of tracking everything

- Too much data!! $\Rightarrow$ Sampling
- Being in the same namespace of your program
- Sharing memory allocation with your program.
- Feeling like I'm a developer on every program team.

Sharing Namespace

- XALT uses xalt_obfuscate.h to hide the xalt library (libxalt_init.so)
- This sharing is common with all libraries
- But usually it is opt-in.
- But very few libraries are uses everywhere, except libc.so

Hiding XALT routine names from users

```
% nm $LD_PRELOAD| grep __XALT_build
0000000000009e80 T __XALT_buildEnvT_xalt_1_5
000000000000a840 T __XALT_buildUserT_xalt_1_5
00000000000163a0 T __XALT_buildXALTRecordT_xalt_1_5
```

- XALT routine names are hidden by macros supplied in xalt_obfuscate.h

Cannot hide routines in system libraries like libuuid

- The routine "**random**" is used by libuuid.so
- A user's program created their own random routine in fortran
- The program crash when XALT was used.
- Solution: use dynamic linking via libdl.so for libuuid.so and others.

User's bug hidden by initially zeroed memory

- Initially all memory is zeroed before program starts
- Note that pointer zero, integer zero and float zero are all zero bits
- Link lists require a NULL pointer at end of list.
- Used memory is *NOT* zeroed for you in C.
- User's program work w/o XALT, Failed with XALT.
- XALT Fix was to zero memory before returning it
- Also only return memory when necessary

XALT has been a group effort

- James McComb & Michael Scott from IU ⇒ Tracking R imports
- Kenneth Hoste ⇒ Riccardo Murri ⇒ Tracking Python imports
- Bill Barth ⇒ Tracking MATLAB imports
- NVIDIA's Scott McMillan ⇒ Tracking NVIDIA GPU usage
- Bilel Hadri ⇒ ALTD at NICS
- Doug James wrote the NSF proposal with help from Mark and me
- See ACKNOWLEDGMENTS.txt for a complete list

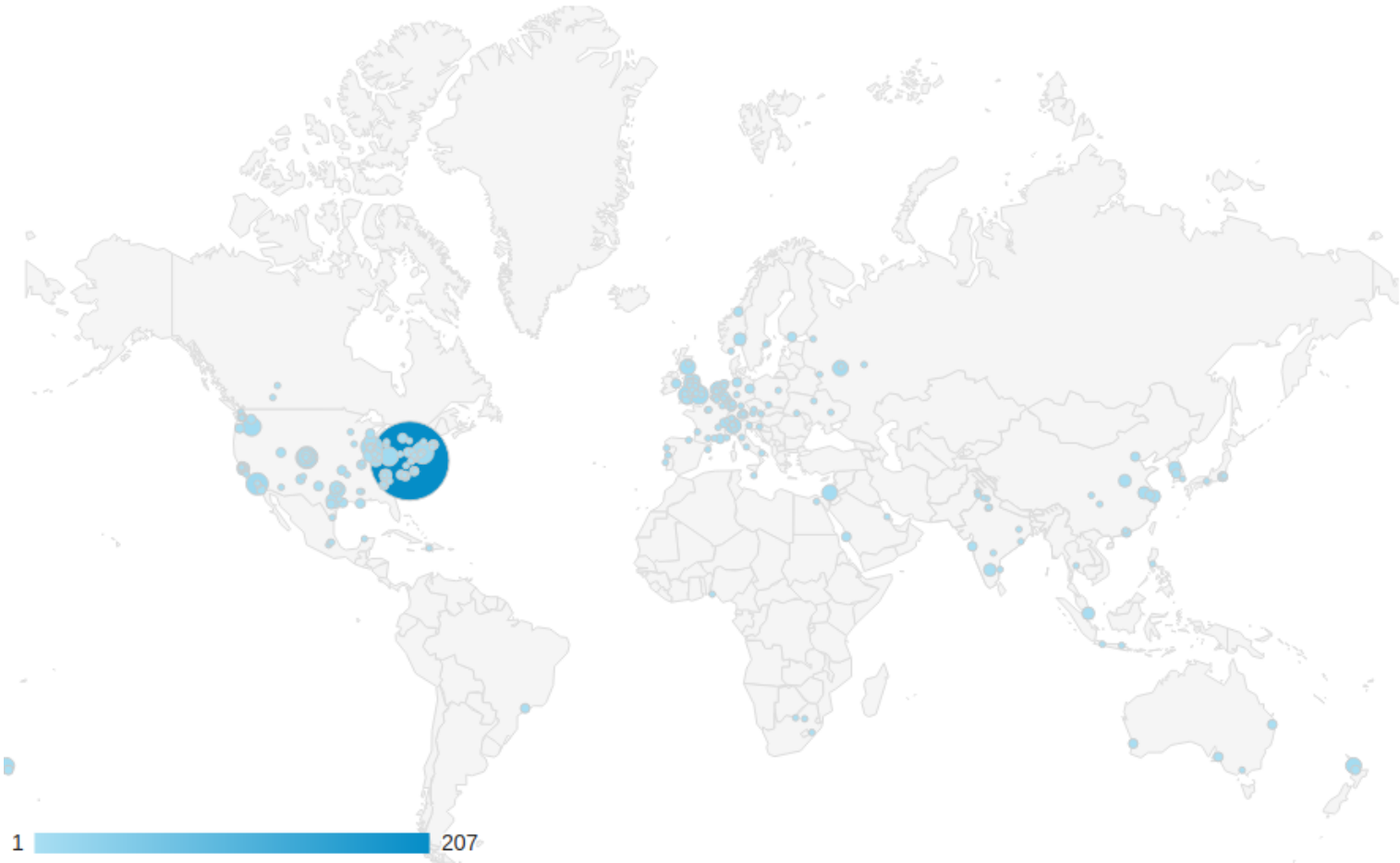
Lessons Learned

- Figuring out how to debug on a remote site via XALT_TRACING=
- Reporting the XALT configuration
- make gittag TAG=...; make world_update
- git worktrees
- Exploiting Lmod to help XALT (reverse map: path $\Rightarrow$ to module)
- Using python config files to build C code.
- Speed, Speed, Speed
- Building trust with the user community

Building trust with the user community

- Making it reliable via Testing
- Timely answering the email and github issues
- Book: Team Geek
- Learning to be polite when answering and re-answering questions
- “You might consider ...”
- “Please test XALT version ... when you get a chance to see if it works for you”
- Not getting upset when non-native English speakers sound insulting

XALT Doc usage by City



Conclusions

- It has been quite the learning experience
- XALT is considerable better for being Open Source

Future Topics?

- Next Meeting will be on September 21, 2023 at 10:00 am U.S. Central (15:00 UTC)
- It will be a different Zoom link as Amit Ruhela will leading the project with some help from me.