- **John Hatcliff** • 15 days ago

  Hi, thanks for listening to my talk. HAMR provides a model-base development layer (using the AADL modeling language) above CAmkES/seL4 - this higher-level abstraction layer aims to better integrate seL4 with MBD system engineering workflows (including a bunch of analysis and verification technologies). HAMR also supports other backends including the JVM and Linux.

  For more about HAMR, you can take a look at the HAMR website http://hamr.sireum.org

  - 1
  - •
  - Reply
  - •
  - Share ›

- 

  John Hatcliff • 15 days ago

  HAMR is being used on several DoD research projects including the DARPA CASE program (we are teamed with Collins Aerospace (lead), Adventium Labs (HAMR tool lead), and Data61), and other SBIRs from AFRL, US Army, and DARPA (these are led by Adventium Labs).

  For more about the Collins CASE vision, see http://loonwerks.com/projec...

  - •
  - Reply
  - •
  - Share ›

- 

  Regan Robertson Mod • 14 days ago

  The video will start on this page at 10:00am, and you may have to hit play or unmute. As a reminder this site works best in a Chrome Browser. You can write in comments through this feature to continue the conversation or ask questions. Please login or sign up for a Disqus account to participate in the discussion boards.

Reply

Share ›

Jason H Li • 14 days ago

from seL4+QEMU testing to seL4+board evaluation, did you see gaps & surprises and how do you handle those?

Reply

Share ›

Todd Carpenter Jason H Li • 14 days ago

Ignoring driver issues, we've had remarkably consistent results moving between an application between QEMU, and a couple different ODroids. This isn't too surprising, considering that the overall model-based development, as well as the tightly constrained communications, eliminates side effects, back channels, unintended control/data flow mixing. What runs on the target is exactly what we specified in the model. This really helps portability.

Until you get to drivers. That's a different issue.

Reply

Share ›

Jason H Li Todd Carpenter • 14 days ago

Thanks Robby and Todd. Timing and drivers - made total sense.

Reply

Share ›

Jason H Li Todd Carpenter • 14 days ago

"What runs on the target is exactly what we specified in the model" - this is asserted test-wise, or you could also put some proofs from spec to what runs on targets. I know this is very hard.

Reply

Share ›

Ihor Kuz Jason H Li • 14 days ago

That's in scope for the CASE project.

Reply

Share ›

Jason H Li Ihor Kuz • 14 days ago

Thanks Ihor. I would imagine so.

Reply

Share ›

Robby Jason H Li • 14 days ago

We observed some differences wrt. timing, especially if the component runs in a (Linux) VM. In such cases, we currently use different schedules for testing on the QEMU vs. boards.

Reply

Share ›

Todd Carpenter Robby • 14 days ago

This timing isn't too surprising - the startup cost of a linux VM is tremendous, especially with emulation, and there is a lot of frontend/backend communications

required. We can address this with different schedules as Robby mentioned, as well as fattening up the schedule (trading off efficiency)

- 1
- Reply
- Share ›

John Hatcliff Jason H Li • 14 days ago • edited

To echo what others have said, here is some more feedback from John Shackleton at Adventium Labs: The biggest difference that I've observed is the timing. Components can run slower or faster on the actual hardware (often faster). Consequently, bottlenecks that appear in QEMU may be different than in the target hardware, or vice versa.

- Reply
- Share ›

Jason H Li John Hatcliff • 14 days ago

Thanks John. That's something I experienced for wireless testing - simulation vs. hardware. Actual timing, bottleneck, performance numbers, etc.

- Reply
- Share ›

Ihor Kuz John Hatcliff • 14 days ago

Which makes perfect sense. Even moving between hardware platforms will affect timing. So QEMU is essentially like a different hardware platform target.

- Reply
- Share ›

Nathaniel Husted • 14 days ago

Would it be possible to get a recorded copy of this video to view later? I've been wanting to learn a bit more about AADL, briefCASE, and the like but probably need to go back and watch things a couple times to fully understand it.

- 2
- Reply
- Share ›

John Hatcliff Nathaniel Husted • 14 days ago

The recorded video is linked from the HAMR home page: http://hamr.sireum.org

- 2
- Reply
- Share ›

Darren Cofer Nathaniel Husted • 14 days ago

There are also a couple of videos showing how to use the BriefCASE tools on an example AADL UAV model: http://loonwerks.com/projec...

1

Reply

Share ›

Robbie VanVossen • 14 days ago

John, we have started trying to utilize HAMR to generate virtualized systems on seL4, but we are running into some issues. We would really be interested in setting up a meeting to work on this more closely.

Reply

Share ›

John Hatcliff Robbie VanVossen • 14 days ago

Sure. We are also rolling out a collection of tutorial videos and text documentation for seL4 development within the next 3-4 weeks. There should be an initial video walkthrough

released by Thanksgiving. Beyond that, we would be happy to coordinate with Collins and Adventium and have some discussions.

Reply

Share ›

Jason H Li John Hatcliff • 14 days ago

John - the CoE will be very interested in tutorials and videos. Maybe we could include a pointer to your stuff on the CoE website. Thoughts?

Reply

Share ›

John Hatcliff Jason H Li • 14 days ago

I would say talk to Darren Cofer and Todd Carpenter about the timing. We would definitely like to continue to raise awareness of HAMR and its applicaiton on CASE, but we may want to get more tutorial and documentation in place, and harden the framework a bit more before really pushing it out to CoE (but talk with Todd, he will have some good insights).

Reply

Share ›

Todd Carpenter John Hatcliff • 14 days ago

I think it's a great idea. We should talk. Let's set up a meeting to discuss logistics, expectations, and timing.

Reply

Share ›

June Andronick • 14 days ago

Nice overview, thanks John!

1

Reply

Share ›

Jerry Dussault • 14 days ago

Very informative John. Thank you!

Reply

Share ›

Renato Levy • 14 days ago

I might have miss that, but does Hamr provides a verification that the protocols within components are complete and secure. How does it enforce it on the implementation of the functionality itself

Reply

Share ›

Renato Levy Renato Levy • 14 days ago

sorry, just to be clear on my question, the API is just a manner in which the components interface. The actual exchange of messages (the protocols) must be complete otherwise, you present opportunities for actions to behave different than expected. At least, this was my previous experience with ASs

Reply

Share ›

John Hatcliff Renato Levy • 14 days ago

Yes, see my answer below, we are trying to take great care with that -- it's a central part of our research. In addition to what I said below, the contract language that is

being rolled out for Slang will eventually allow is to be able to prove key properties of the infrastructure code itself (being written mostly in Slang). Also, something that I didn't mention in the talk is that the C generated from Slang is compatible with the CompCert verified C compiler, so connecting with the CompCert assurance story is also part of our long-term objectives.

Reply

Share ›

John Hatcliff Renato Levy • 14 days ago

The short answer is that right now there are no "proofs" but that the HAMR architecture is set up to enable that in the future. The framework encodes most of the important logic for the communication and threading in Slang, which can be both run on the JVM and translated to C. That ensures that the same "specification" (i.e., coding in a rather high-level language) is used to generate the infrastructure code across the different platforms. Right now we are working on an operational semantics for the infrastructure, to support further verification efforts. We are also setting up testing infrastructure that will enable different platform infrastructure to be tested against the reference semantics in Slang. You can read more about the overall strategy in this link: http://hamr.sireum.org/hamr...

Reply

Share ›

Renato Levy John Hatcliff • 14 days ago

I did work before on validating the protocols between components to guarantee its correctness and completeness. This was done as part of a verification on Autonomous agents behaviors, but i see no reason why it cannot be applied here.

Reply

Share ›

John Hatcliff Renato Levy • 14 days ago

Yes, that's what we are working on. But the first priority is to get the semantics (target of the verification) finalized and the contract verification framework in place (see also my comments to Jason above).

Reply

Share ›

Jason H Li • 14 days ago • edited

I also have an old-and-dumb question (did some code gen myself in the 90s) .... how do you know that your generated code is correct? ... well, in some good sense would be enough I guess.

Reply

Share ›

Robby Jason H Li • 14 days ago

This is actually the question that should be asked for anyone claiming high assurance code generation. Please see John's answers to Renato's questions that summarize our strategies.

- 1
- Reply
- Share ›

Jason H Li Robby • 14 days ago

Yes just saw that, and would agree with the strategies. Pretty much the best we can do now.

- Reply
- Share ›

John Hatcliff Jason H Li • 14 days ago

No, that is a great question. See my comments to Renato below regarding the infrastructure code. For the application code, we have a DARPA SBIR from Ray Richards with Adventium Labs where we are working on a contract verification framework for Slang that connects to contracts specified at the AADL. If you are familiar with Spark Ada, the framework is similar to that -- but for Slang (Scala subset). You can use the Slang verification framework to prove (SMT-based) that your code conforms to contracts (again

aligned with AADL semantics and component boundaries). Furthermore, you can use the framework to prove the that integration / composition of components meets system level contracts. Then application code code correctness (discussed in this post) + infrastructure code correctness (building out on the comments to Renato) + correctness of CAmkES / seL4 gives you your overall correctness / assurance story.

- 1
- Reply
- Share ›

Jason H Li John Hatcliff • 14 days ago

Love this entire story! Maybe we should follow up with further discussions on these. Will follow up via emails.

- Reply
- Share ›

Regan Robertson Mod • 14 days ago

We have a 30-minute break right now. The next presentation starts at 11:00 AM.

- Reply
- Share ›

Jason H Li • 14 days ago

BTW - love this talk! Triggered many thoughts.

- 1
- •
- Reply
- •
- Share ›

John Hatcliff Jason H Li • 14 days ago

Super! We would be happy to discuss more. I think it connects to a number of things that Sergey and Bill Scherlis were mentioning about developing high-level abstractions that can be reasoned about earlier in the design process with model-based techniques -- then ensuring that those abstractions are faithfully deployed on platforms.

- Reply
- Share ›

—

Jason H Li John Hatcliff • 14 days ago

Totally John. A better design earlier in the cycle would eliminate so many bad stuff down the road.