- **Regan Robertson** Mod • 15 days ago
  Good Morning,
  The video will start on this page at 11:00am, 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.
  - Reply
  - Share ›

Stuart Card • 15 days ago
One idea with which we have been toying is running x86, ARM & RISC-V processors in parallel: while this won't save us from bugs at the source code (or higher) level, it should help with compiler and below (including CPU specific hardware vulnerabilities).
- 1
- Reply
- Share ›

Douglas Schafer Stuart Card • 15 days ago
Hey Stu, kind of like popcorn-Linux idea?
- 2
- Reply
- Share ›

Stuart Card Douglas Schafer • 15 days ago

"kind of" like their HEXO, yes, details of our approach as yet TBD. :-)

Reply

Share ›

Ryan Craven Douglas Schafer • 15 days ago

+1 for mention of popcorn. Popcorn is more general-purpose while the stuff in my talk is geared to CPS.

Link to learn more about separate Popcorn topic: http://www.popcornlinux.org...

Reply

Share ›

Ryan Craven Stuart Card • 15 days ago

It could still help with higher bugs. Yes, the bug would be present in all architectures, but the input the attacker is forced to craft to exploit that bug is different across those replicas. Depends on what the goal is.

- Reply
- Share ›

Jerry Dussault Stuart Card • 15 days ago • edited

How would you detect that the running systems have diverged, and determine which one has been compromised? Very interesting to think about!

- Reply
- Share ›

Ryan Craven Jerry Dussault • 15 days ago

You need watchdog timers, or some form of an orchestration system. Many existing fault tolerance / hot backup systems already have methods for doing this. Another nice synergy with integrating these techniques into existing fault tolerant systems.

- 1
- Reply
- Share ›

Renato Levy Jerry Dussault • 15 days ago

you diverge, but you cannot loose control, either crash or be owned. If you can recover in time, cool!

- Reply
- Share ›

Stuart Card Jerry Dussault • 15 days ago

In addition to software based methods as cited in Ryan Craven's reply, we are considering hardware based comparison of outputs: when they are discrete control lines (common in embedded applications), this is fairly straightforward.

- Reply
- Share ›

Jerry Dussault Stuart Card • 15 days ago

Comparison sounds reasonable, but does that require > 2 parallel copies so you can identify the compromised processor?

- Reply
- Share ›

Stuart Card Jerry Dussault • 15 days ago

Minimum of 3 for simple voting. With only 2, I don't see how to do it in real time, except when one exceeds fixed bounds on allowed output values.

Reply

Share ›

Ryan Craven Stuart Card • 15 days ago

One potential issue with comparison. If you're doing general-purpose computing it could get quite a bit more complex. i.e., what are the various "equivalence points"? that depends on the process being executed and you have to extract some structure from it. You'd also have to define some cut-off time (e.g., what if one of your variants is correct, but just takes longer? that's not an attack but just a slow replica).

CPS makes this easier because we have the periodic scan cycle happening ad infinitum. The cycle also has to meet real-time deadlines that are set by the physical system requirements. That gives good structure for comparison.

Reply

Share ›

Jason H Li • 15 days ago • edited

slide 11 - I would say a good manager / policy engine is needed to coordinate and manage these in mission / platform context. For defeating these common node failures.

Reply

Share ›

Renato Levy • 15 days ago

How do you figure out that you are owned. A good attack would mask this

Reply

Share ›

Jason H Li Renato Levy • 15 days ago

This goes back to something we know/are but the bad guys don't know / are not. IMHO.

Reply

Share ›

Ryan Craven Renato Levy • 15 days ago

The diversification is key to discovering this. In the example in the slides, if we had run C0 by itself and it gets owned and we never know it because it's still providing inputs on time (we have no idea they're attacker-controlled). but by having the diverse replicas, we infer that someone has been owned when we see non-uniform crashes happen across the replicas.

Reply

Share ›

Jason H Li • 15 days ago

The inertia is a boon allowing us to monitor and check. Like the idea of synergizing with fault tolerance ideas. Slides 17 and 18 are very enlightening. Would love to see some of these happen in real platforms, in some practical holistic manner, well tested and orchestrated in context.

1

Reply

Share ›

Ryan Craven Jason H Li • 15 days ago

Thanks Jason! That's what we're working toward. Getting into real platforms.

- Reply
- Share ›

Jerry Dussault • 15 days ago

Thanks Dr. Craven, nice presentation!

- 1
- Reply
- Share ›

Regan Robertson Mod • 15 days ago

Please join us at the next session that has started. You can either go back to agenda to get to the next session or at the bottom of the page there is a next session button.

- Reply
- Share ›