Rebuttals to >Oddness of JALR

Rebuttal:
The “student” has misconstrued the concepts:

A) mepc does not avoid the base plus offset odd-odd advance.
A service routine calculating the instruction result WILL also
add the two low bits and advance the target address.
All that mepc will do is the final step in the jalr – to clear the low bit.
Note: this was obfuscated in the “paper” with the digression to inconsistent
behaviour IALIGN=32 vs 16 – which leads to the next point:

B) mepc behaviour and jalr are intentionally different, and justifiably.
i) We add a level of validation for code using jalr in align=32 code.
The user level code can benefit from this discrepancy because it
will trap and cause a alert that the user code has misformed the
critical jump address. There is a real value to provide such checking.
ii) mepc use on the other hand is a shorthand for system code that
is generally well vetted. M-mode can explicitly check for misaligned
return address. But whether it does or not, it can save at least one instruction
in the critical code sections.
But more importantly than that, removing a potential trap from mepc reduces
the hardware complexity in what could be on the critical timing path.
Avoiding this for special instructions is prudent.

Response:
Yes, all true and justifiable, but is it best?

IALIGN=32 hardware can legitimately remove this exception, noting that M-mode handler can always be written to ignore it and thus it becomes an implicit feature of the m-mode handler.

So, we can debate this decision on the merits, and time will tell which is the optimal hardware solution.

But it ignores the true point:
Students of RISCV are inclined to misunderstand. It is human. It is inherent in our natures.
AND it is such flawed individuals that will soon be writing code that ignores the danger of JALR.

Rebuttal:

The Black Hat section is speculative, lacking in rigor.
Even the contrived example of a LTO module that will setup a situation that the compiler will naturally avoid is farfetched. By its own admission, the scenario depicts a future state with an advanced branch instruction that has not officially been proposed to any TG or SIG body. Even then, it is not the new state of the art that it targets but those who wish comparable features in the RVC base domain.
The whole body of IALIGN=32 are immune to the attack. The concurrence of events required to make this scenario viable is highly unlikely. There is no systematic method existing that predisposes to this preposterous situation, so widespread exposure is diminutive to the point that it does not merit the effort [and confusion/offsetting effects] to attempt some kind of mitigation.

Response:

The situation I propose is arguably weak, contrived, speculative and deficient in many salient aspects that would be required to weaponize odd-odd JALR. It does not rise to the bar of CVE submission.

I am not a Black Hat. I know I am not as smart as many of them individually, and certainly not collectively. However, if even I could come up with a means, as contrived as it might be,
to infiltrate code with odd offset jalr, how much more the professions for whom this is there livelihood and, for many, their only means of raising themselves and their families out of poverty. The situation ripe for Black Hat's is due to inequity, socio-economic and institutional. We at RISC.org are not immune to such inequity. We may hide behind "meritocracy" but we cannot reasonably believe ourselves wise enough and sufficiently impartial to be even reasonably confident that we operate equitably or even benevolently.
We can, however,contribute to a world that is more equitable in being honest ourselves and by avoiding opportunites for others to see an easly way through deception. Depreciating odd-offset JALR is for me such an occasion to: do the right thing and believe that God will bless the results of it.