The Hyperscan API makes the use of specialized instruction sets relatively straightforward. It provides `hs_populate_platform()` which returns an `hs_platform_info`, which can in turn be fed into the various `hs_compile` calls.

It looks like the HS module in Suricata is currently feeding NULL into these functions eg: `hs_compile_multi()`

```c
err = hs_compile_ext_multi((const char *)cd->expressions, cd->flags, 
    cd->ids, (const hs_expr_ext_t *)cd->ext, 
    cd->pattern_cnt, HS_MODE_BLOCK, NULL, &pd->hs_db, 
    &compile_err);
```

I'm no C programmer, but this looks like an improvement I should be able to add with context from issue 2010. I've assigned it to myself for now and will see if I can get it going. Reading over this presentation makes it sound like AVX2/512 will be beneficial.

Please advise if this is obviously more complicated than it appears to me.

---

### History

**#1 - 12/14/2018 03:45 PM - booble tins**

booble tins wrote:

> Please advise if this is obviously more complicated than it appears to me.

Significantly less complicated than expected. From: https://intel.github.io/hyperscan/dev-reference/compilation.html#instr-specialization

> If this argument is NULL, the database will be targeted at the current host platform.

I missed that on the first read, apologies for the waste of reading time!

**#2 - 12/14/2018 03:49 PM - Victor Julien**

Maybe it would be nice to see if we can print the used algo/instruction set to the user?

**#3 - 06/15/2019 09:52 PM - Andreas Herz**

- Target version set to TBD