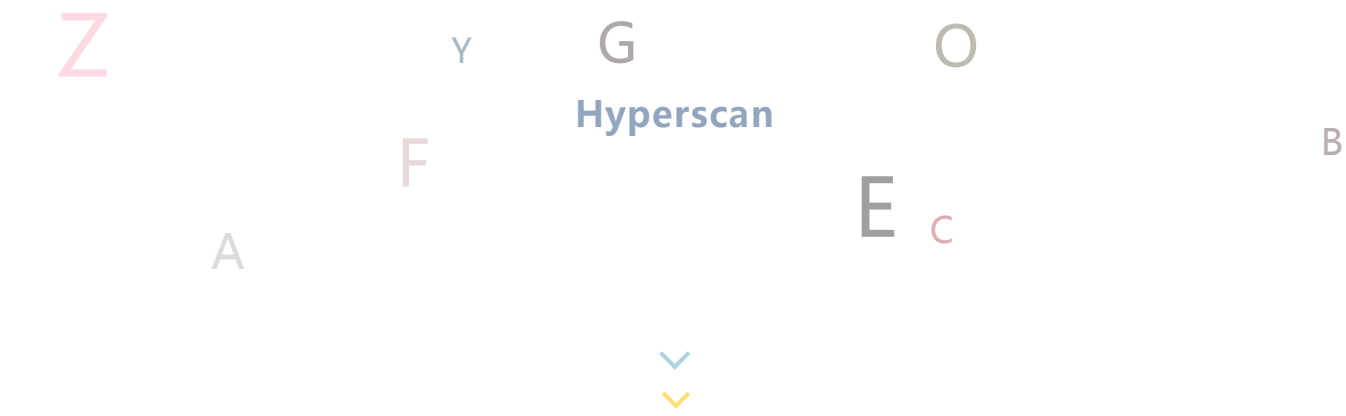
Hyperscan 常见问题解答

原创 DPDK开源社区 2017-08-25

作者 DPDK开源社区



对于大家热切关心的有关Hyperscan的一些问题，我们进行了整理，并在文章内做出了以下解答：

运行期相关问题

01

Q: Hyperscan流模式中的hs_open_stream() 和hs_close_stream()对性能的影响有多大？

A：在流模式中，Hyperscan只需要在扫描开始前使用hs_open_stream() 对bytecode进行流模式的预处理，并在最后一条流扫描完毕后调用hs_close_stream() 进行尾部锚点的匹配以及流的关闭操作。因此，流的开启和关闭操作在Hyperscan运行期只占了很小的时间比例。

02

Q: Intel处理器的SIMD指令集对Hyperscan性能提升有多大的帮助？

A: Hyperscan内部的多个引擎都利用了SIMD宽指令集的加速优化。例如在字符串匹配引擎中，我们利用SIMD指令对一长段输入数据进行批量的查询匹配。指令集支持的宽度越宽，匹配速度提升的就越明显。因此，在新的平台上Hyperscan基于更新的指令集，可以发挥更高的性能。

03

Q: Start of Match (SoM)会影响Hyperscan性能吗？

A: 开启SoM之后，Hyperscan会做许多额外的工作用于SoM的计算。对SoM的计算是昂贵的，它同时也需要消耗更多的流状态空间。在轻量级的测试数据下，SoM对Hyperscan的性能影响约为10%。因此我们建议用户在非必要时不要开启SoM功能。

04

Q: 将一集规则分成多组分别编译并扫描，与统一编译扫描相比，哪一个效率更高？

A: 通常情况下，将所有规则统一编译会得到更高的效率。因为Hyperscan编译期会将所有规则转化成一张大的NFA图，期间相似的规则会被大幅地合并，因此运行时效率更高。
但特殊情况下，将一部分特殊的规则进行单独编译是更好的选择。例如，用户对模式的使用需求为：绝大部分规则都只会用在块模式中，但个别规则需要在流模式中扫描，那么将一小部分流模式的规则单独编译，将更有利于大多数块模式规则优化，提高运行期性能。

05

编译期相关的问题：

06

Q: Hyperscan编译后生成的数据库是怎么样的规模？以及数据库大小与规则条数间存在着怎么样的关系？

A: 通常来说规则条数越多，数据库自然越大，但两者间并不是一个线性的关系。单条正则表达式规则编译出的数据库大小在1KB~5KB的级别，而一个包含一千条类似规则的签名编译出的数据库规模大约在100KB级别。
影响数据库大小的更重要因素是规则的复杂程度。较简单的纯字符串规则比复杂的正则表达式规则消耗更少的空间。

07

Q: 通常Hyperscan的数据库需要编译多久？

A: 对于轻量级(1K条以下)的规则集，数据库的编译时间一般在1s以内。我们也对更大规模的规则集做过测试，对于规模为几万条的规则集，编译时间通常在10s以内。

08

Q: 当重新编译生成新的数据库后，跟数据库相对应的scratch也要进行切换吗？

A: Hyperscan提供了相应的API来进行scratch的各类操作。接口hs_alloc_scratch()用于scratch空间的分配以及替换。对于用户提供的新的数据库以及旧scratch，Hyperscan会先行验证旧的scratch空间是否满足新数据库的使用需求，如果全部满足则不需要替换，否则自动为scratch分配更大空间。此外接口hs_clone_scratch()则可以帮助用户将同一scratch快速拷贝，避免再次通过数据库进行生成。

09

Q: Hyperscan编译期的峰值内存占用情况如何？与规则条数之间存在着怎样的关系？

A: 对于较少的规则数量，Hyperscan编译期的峰值内存消耗约在30MB的规模；当规则数量较大时(10K条以上)，峰值内存消耗会上升至100MB。

10

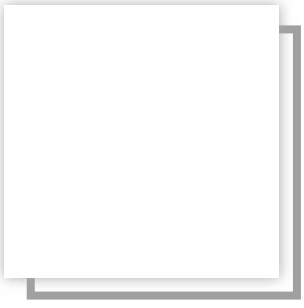
Q: Hyperscan支持动态编译（增量编译）吗？

A: 由于Hyperscan编译期生成的数据库对输入的匹配规则进行了大量的优化，因此当规则集中新增了规则时，Hyperscan需要重新编译出新的NFA图并做相应优化。目前不支持增量编译。

往期文章精选

- DPDK Release 17.08
- Hyperscan中的 NFA模型演化
- Hsbench的Hyperscan性能分析
- DPDK报文处理框架简介
- Hyperscan在Suricata中的应用
- Intel 82599ES 基于DPDK和Linux kernel的L2fwd性能比较
- 25G来了，你准备好了吗？
- Hyperscan Release 4.5.0
- Virtio 零丢包测试配置优化
- 探秘DPDK Virtio的不同路径，so easy!
- 用TestPMD测试DPDK性能和功能
- VFD大揭秘，一定有你想知道的！
- cryptodev 概述、状态及未来的工作——DPDK用户空间大会
- 从PCRE到Hyperscan
- Generic Flow API简介
- Hyperscan与Snort的集成方案
- DPDK开源社区加入Linux基金会大家庭了
- DPDK中的memcpy性能优化及思考
- Hyperscan的模式选择

长按扫描二维码关注我们



投诉