Hyperscan_in_Rspamd

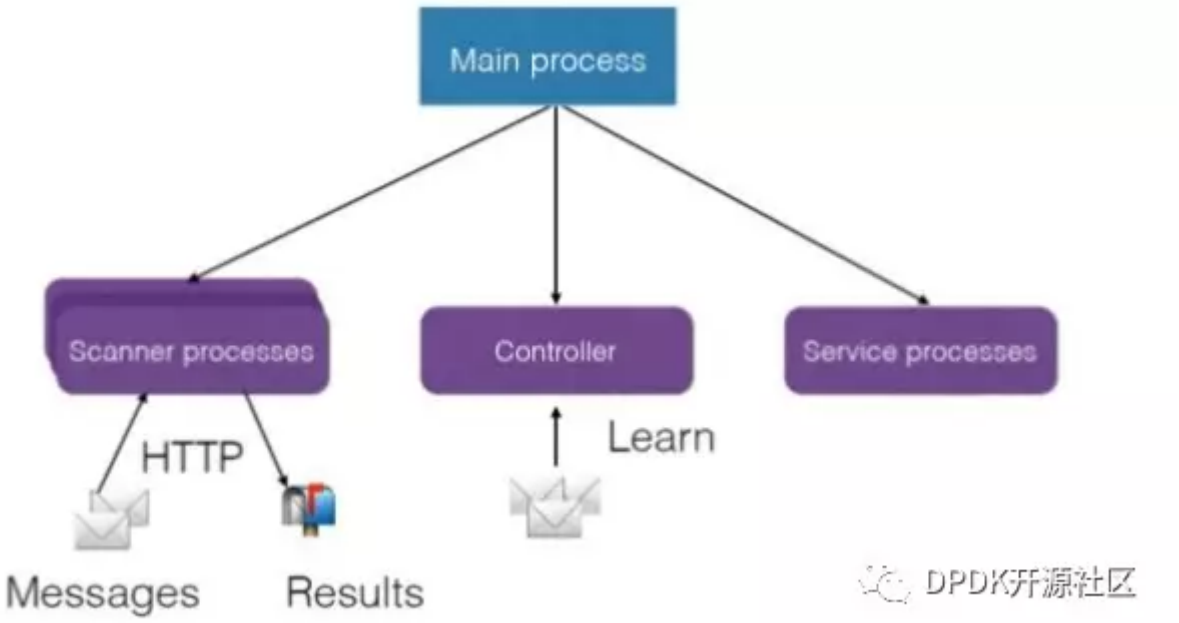
原创 DPDK开源社区 2017-09-01

作者 昌昊

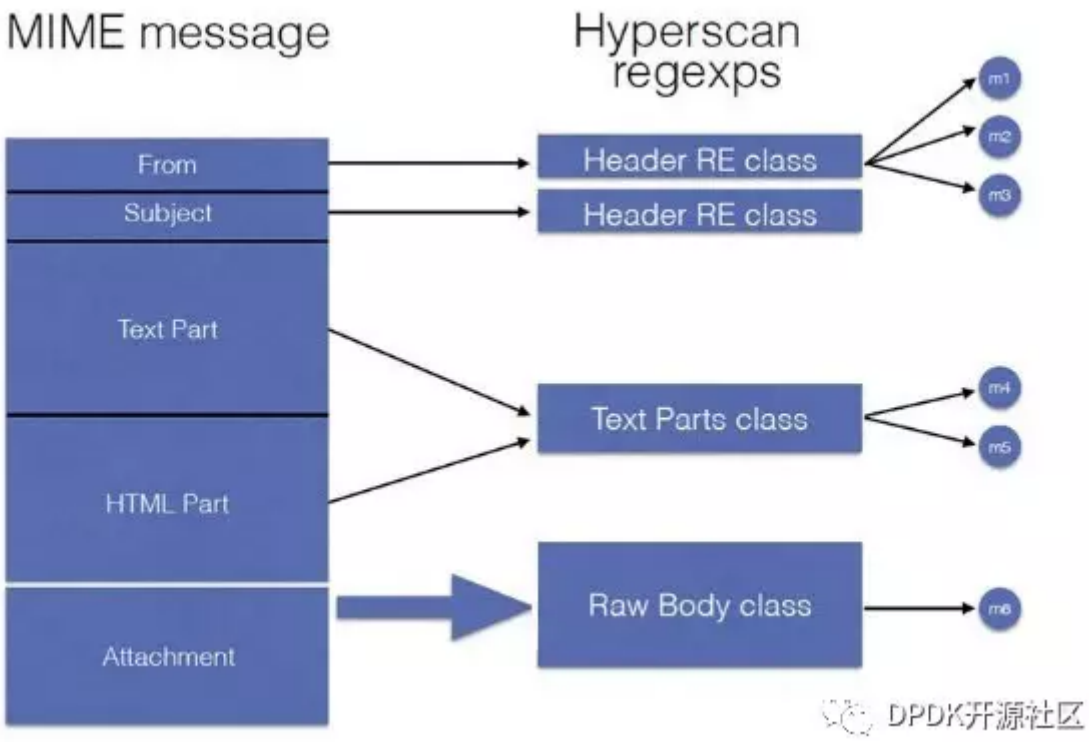
之前我们介绍过Hyperscan在IDS/IPS解决方案Snort、Suricata中的应用。本文将介绍Hyperscan在电子邮件过滤系统Rspamd中的应用。

Rspamd简介

Rspamd(https://www.rspamd.com)是一款快速、免费、开源的电子邮件过滤系统，由主进程、扫描进程、控制器和服务进程等部分构成。



电子邮件过滤系统长期以来一直是模式匹配技术的重要用户，通常要求扫描通过邮件服务器的每个消息，对大量的正则表达式规则集进行匹配。Rspamd也不例外，每条MIME(Multipurpose Internet Mail Extensions)消息的各个部分都对应有预定义的正则规则集，Rspamd扫描消息各部分来检测是否有对应正则规则集上的匹配。原生的Rspamd使用PCRE做正则表达式匹配工作，但PCRE只能在一次扫描中对单条正则表达式进行匹配，对于大规模正则规则集需要对相同输入进行多次扫描。



Hyperscan在Rspamd中的应用

自Rspamd 1.1 release开始，将Hyperscan集成其中，用于处理正则表达式匹配任务。其关键点在于下述几个方面：

高性能

Hyperscan在单条正则表达式匹配上的表现比PCRE更加优秀。此前有专门的文章(从PCRE到Hyperscan)介绍对比测试的结果。

多模式匹配

一种简单的解决方案是将多条正则规则用“或”合并成单条，例如：/abc/, /cde/, /efg/ 三条规则可以合并为 /(abc)|(cde)|(efg)/

但并非所有情况都可以这样做。通常我们的正则表达式是带有标志的，例如：

/foo.*bar/H – 单次匹配

/[a-f]{6,10}/i – 忽略大小写

/^GET\s.*HTTP/m – 支持换行

这些正则表达式无法简单的用”或”合并，但对于Hyperscan而言并不是问题。

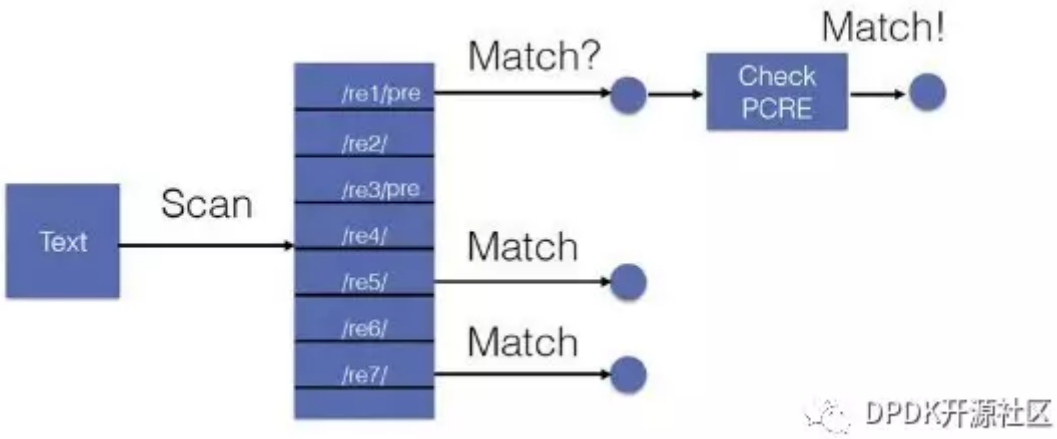
多模式匹配(MPM: Multi-Pattern Matching)是Hyperscan的一大优势，Hyperscan支持对多条正则规则进行一次编译以及对输入数据进行一次扫描便找出其中所有的匹配。相比PCRE，这将大幅减少所需扫描的数据量，且在性能方面更优于对每条正则规则进行逐次扫描的累加结果。(参考：从PCRE到Hyperscan)

预过滤模式支持

目前Hyperscan不能全面覆盖PCRE的语法，不支持反向引用(Back Reference)和零宽断言(Zero-Width Assertion)。但由于Hyperscan相对PCRE的性能优势，可以将这类Hyperscan不支持的语法转化为一个超集且为Hyperscan支持的语法做预过滤匹配。例如：

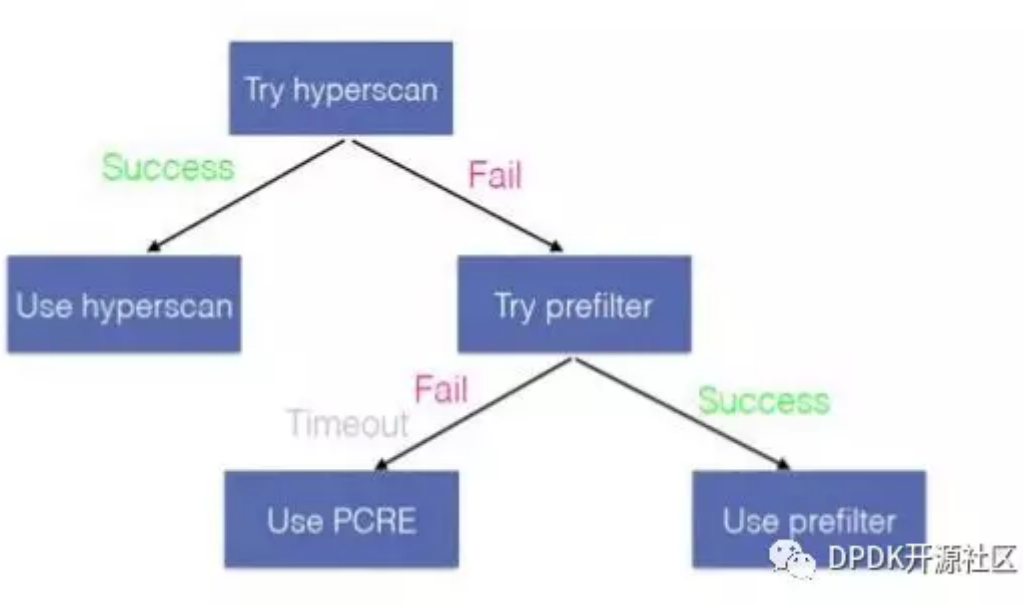
/foo(\d)+bar\1baz/ 转化为 /foo(\d)+bar(\d)+baz/

转化后的正则表达式与未经转化的正则表达式一起编译，对输入数据进行扫描，产生数处匹配。对正常匹配直接报告；对预过滤产生的匹配用PCRE做一次确认，确认成功才报告。如下图：



Hyperscan预过滤模式使用

Rspamd中对正则规则集中每条正则表达式做检查，确认是否为Hyperscan所支持，或者是否可使用Hyperscan预过滤功能。流程如下图所示：



各阶段是调用什么API实现的呢？下面简要说明：

Try Hyperscan

```
hs_compile(pats[i], flags[i], ...)
```

Use Hyperscan

```
hs_pats[n] = pats[i]
hs_flags[n] = flags[i]
hs_ids[n++] = i
```

Try Prefilter

```
hs_compile(pats[i], flags[i] | HS_FLAG_PREFILTER, ...)
```

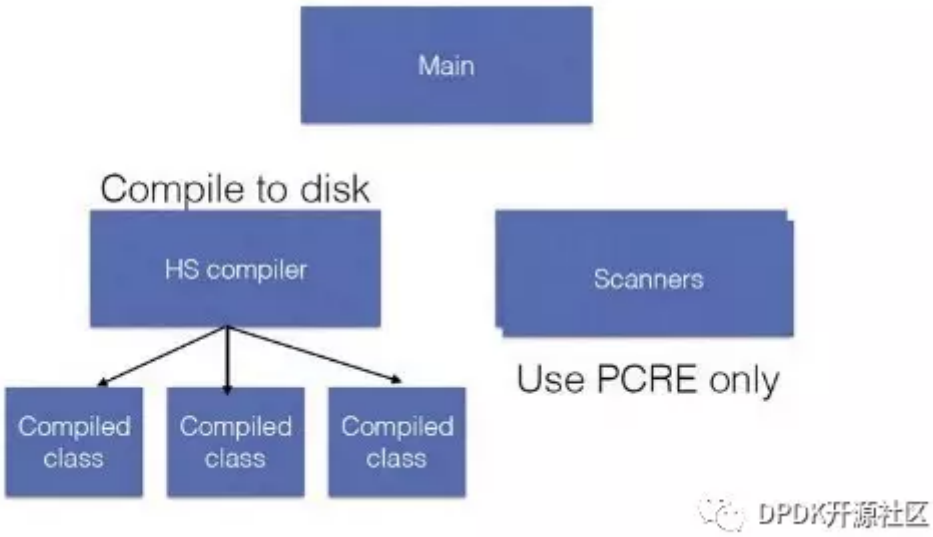
Use Prefilter

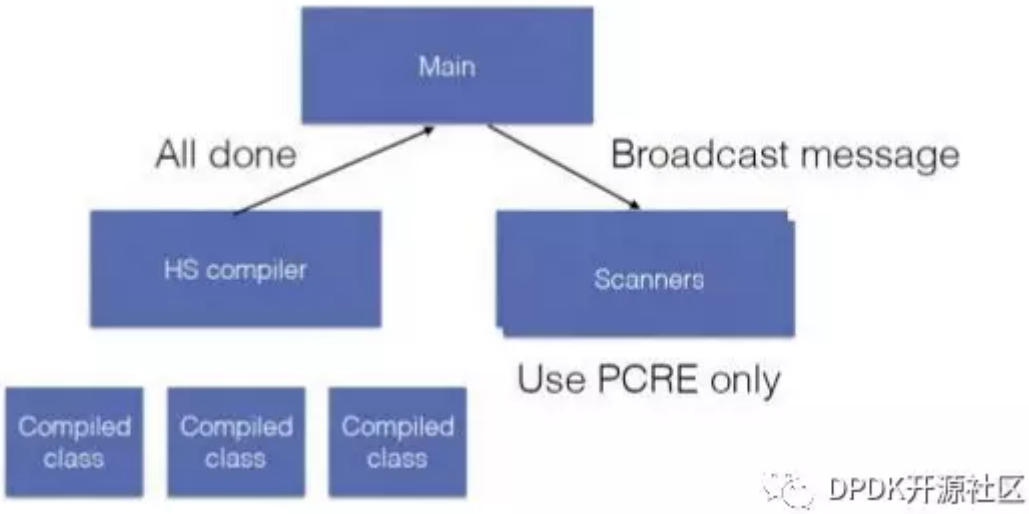
```
hs_pats[n] = pats[i]
hs_flags[n] = flags[i] | HS_FLAG_PREFILTER
```

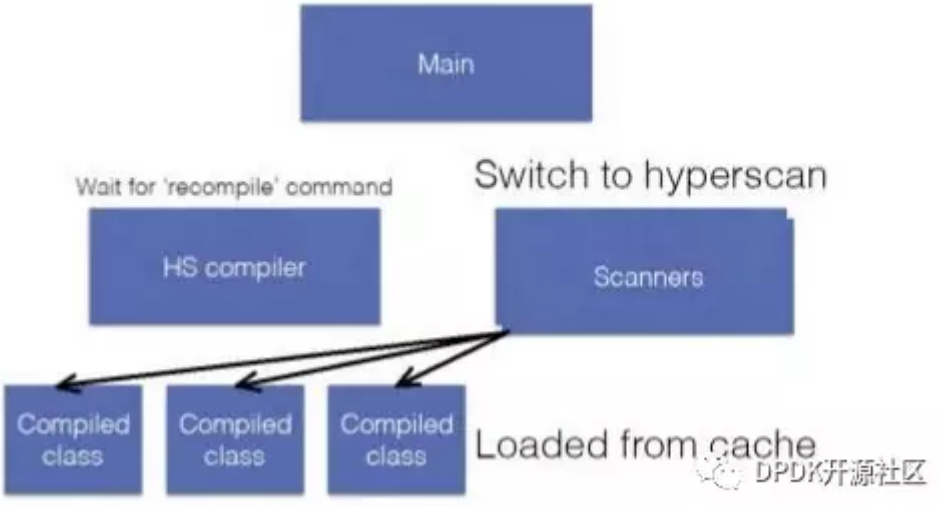
hs_ids[n++] = i

其中pats[], flags[]数组保存整个正则规则集及其标志。Hyperscan或预过滤支持的正则表达式及其标志和id被保存在hs_pats[], hs_flags[], hs_ids[]数组中，使用预过滤功能的正则表达式在对应标志中加入了HS_FLAG_PREFILTER标志位。

准备工作完成后，Hyperscan对hs_pats[]中的所有正则表达式统一进行编译及序列化，由扫描进程读取并反序列化后使用，如下列图所示：







其编译期和运行期所调用的API为：

HS Compiler（可左右滑动）

```
hs_compile_multi(hs_pats, hs_flags, hs_ids, n, ..., &hs_db, ...)
hs_serialize_database(hs_db, &hs_serialized, &serialized_len)
```

Scanner（可左右滑动）

```
hs_deserialize_database(ptr, len, &hs_db)
hs_alloc_scratch(hs_db, &hs_scratch)
hs_scan(hs_db, data, len, 0, hs_scratch, cb, cb_data)
```

性能数据

下面是一组来自于Rspamd作者Vsevolod Stakhov提供的测试数据，对于相同长度的消息，原生Rspamd与集成Hyperscan的Rspamd在扫描匹配用时、扫描数据量等方面进行了比较：

原生Rspamd(只使用PCRE):

len: 610591, time: 882.251ms
regexp statistics:
4095 pcre regexps scanned,
694M bytes scanned using pcre

集成Hyperscan的Rspamd(Hyperscan + Prefilter + PCRE):

len: 610591, time: 309.785ms
regexp statistics:
34 pcre regexps scanned,
8.41M bytes scanned using pcre,
9.56M bytes scanned total

详情可参考以下文档：
https://www.slideshare.net/VsevolodStakhov/rspamdhyperscan

作者简介

昌昊：英特尔软件工程师，负责Hyperscan算法开发和性能调优相关工作。主要研究领域包括自动机与正则表达式匹配等。

DPDK开源社区 文章精选

欢迎搭乘Hyperscan号极速列车
从PCRE到Hyperscan

Hyperscan与Snort的集成方案
FD.IO/VPP和DPDK Cryptodev，会产生什么样的化学反应？
DPDK Release 17.05
探秘DPDK Virtio的不同路径，so easy!
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
Generic Flow API简介
无锁队列详细分解——Lock与Cache，到底有没有锁？
从计算机架构师的角度看DPDK性能
无锁队列详细分解 — 顶层设计
VMware Player 搭建DPDK实验平台
Qemu/Kvm 搭建DPDK实验平台
技术贴：利用DPDK加速容器网络
DPDK IP分片与重组设计实现
DPDK流分类优化，不得不知道的事~

DPDK开源社区

最权威的DPDK社区

长按二维码关注

阅读原文 投诉