High-Throughput Distributed Indexing: Scaling Keyword Search on the Linux Kernel Mailing List (LKML)

John Rathgeber, Sean Kim, Anish Dharam, Christopher Zou
Brown University, Providence, RI 02912

Background

LKML is the public email archive for Linux development

What an LKML page looks like:

```
linux-kernel.vger.kernel.org archive mirror
 search  help / color / mirror / Atom feed

* [PATCH ath-next 0/2] wifi: ath12k: Consistently name struct ath12k_base pointers
@ 2026-04-09 18:44 Jeff Johnson
  2026-04-09 18:44 ` [PATCH ath-next 1/2] wifi: ath12k: Fix HTC prototype ath12k_base parameters Jeff Johnson
                    ` (3 more replies)
  0 siblings, 4 replies; 5+ messages in thread
From: Jeff Johnson @ 2026-04-09 18:44 UTC (permalink / raw)
  To: Jeff Johnson; +Cc: linux-wireless, ath12k, linux-kernel, Jeff Johnson

Per ath12k convention, a pointer to struct ath12k_base should be named
'ab', but there are a few places it is named 'ar', so fix them.

Note that one instance in ath12k_wmi_tlv_parse() is not modified since
that instance is being removed as part of:

https://patch.msgid.link/20260407095426.3285574-1-nico.escande@gmail.com/

---
Jeff Johnson (2):
      wifi: ath12k: Fix HTC prototype ath12k_base parameters
      wifi: ath12k: Fix ath12k_dp_htt_tlv_iter()'s iter() signature

 drivers/net/wireless/ath/ath12k/dp_htt.c | 2 +-
 drivers/net/wireless/ath/ath12k/dp_htt.h | 2 +-
 drivers/net/wireless/ath/ath12k/htc.h    | 8 ++++----
 3 files changed, 6 insertions(+), 6 deletions(-)
---
base-commit: 15551ababf6d4e857f2101366a0c3eaa86dd822c
change-id: 20260403-ath12k-htc-proto-9cdc961f39dc

^ permalink raw reply   [flat|nested] 5+ messages in thread
```

Our Goal: Find the most relevant LKML pages given a user-inputted keyword

Google

Similar to Google, except searching the LKML files instead of the whole internet!

Our Approach

Deployment
- Three t3.medium 20GB EC2 workers, one coordinator
- Three distributed groups: frontier, corpus, index
- Consistent hashing decides every key's owner

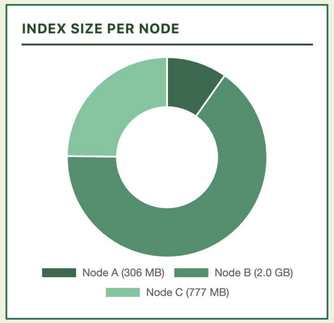
Amazon EC2

Crawling
- Workers fan out, coord dedupes
- Links re-sharded through frontier group
- Stops at ~100K visited URLs
- Axios header to get around Anubis bot-blocker

```
https.get(u, {headers: {'User-Agent': 'axios/1.6.0'},
```

Indexing
- Runs as one MapReduce job
- Map: shell pipeline tokenizes and stems
- Generate unigrams, bigrams, and trigrams
- Shuffle routes terms to owning nodes
- Reduce writes directly into final index

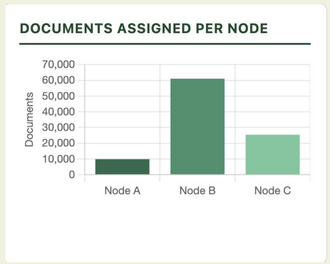
Query
- Same shell pipeline stems input
- Parallel lookups, one per ngram
- Scores summed, top 10 returned



Limitations

Partitioning behavior
- In principle, hashing should balance keys roughly evenly.
- In our run, distribution was still uneven: about 61K / 25K / 10K docs across nodes.

Pipeline constraints
- Ranking is TF-style; no BM25 or semantic reranking yet.
- NLP is expensive (execSync + shell pipeline per document), which dominates indexing time.
- Current deployment is 3 nodes, which limits scale and fault tolerance.
- Coordinator is still a central point for orchestration.
- Uneven shard load can create stragglers and storage imbalance.



Results

Performance

Crawl (~4 mins) → **Index** (~8 hours) → **Query** (~1.96 ms/req)

Corpus Size	Unique Terms
96,388 — LKML documents indexed	~759,000 — unigrams, bigrams, and trigrams
Total Index Size	**Indexing Time**
3.1 GB — distributed across 3 nodes	~8 hours — 3.35 docs/sec, bottlenecked by NLP pipeline

Query Latency
~1.96 ms average across 200 requests to 3 distributed EC2 nodes

Query Throughput
~505.8 queries/sec, measured via sequential single-term index lookups

Correctness

Search for “drive” keyword:

```
ubuntu@ip-172-31-10-12:~/cs1380$ node query.js 172.31.10.12 drive
1. https://lore.kernel.org/lkml/20260323-artificial-ecstatic-collie-047169-mkl@pengutronix.de/t/#u (0.0011)
2. https://lore.kernel.org/lkml/20260331-shaggy-blond-weasel-1bf32b-mkl@pengutronix.de/T/#u (0.0011)
3. https://lore.kernel.org/lkml/d145f868-e3ae-4db1-b85b-9d5756d8fa05@suse.com/#R (0.0006)
4. https://lore.kernel.org/lkml/20260330224619.2620782-1-paulk@sys-base.io/T/#t (0.0004)
5. https://lore.kernel.org/lkml/9392fea00a9c3b23d1bc9468faa1b3cc20904398.camel@gmail.com/T/#m5728482511472977b7bbe4af2fac70e8a4b9c1d4 (0.0003)
6. https://lore.kernel.org/lkml/9392fea00a9c3b23d1bc9468faa1b3cc20904398.camel@gmail.com/T/#m04a6e6cd1fb0521665392a577178af3b9258fc9d (0.0003)
7. https://lore.kernel.org/lkml/moefzqwp3srzpewsuhecxp22cheuplvdhacct5hejzneukdnmu@cchguntsnto4/T/#me410da51cc0852b2f09d8e2eb186b13238fb7449 (0.0002)
8. https://lore.kernel.org/lkml/moefzqwp3srzpewsuhecxp22cheuplvdhacct5hejzneukdnmu@cchguntsnto4/T/#u (0.0002)
9. https://lore.kernel.org/lkml/acay1yaBDXdjNmtj@gpd4/T/#u (0.0002)
10. https://lore.kernel.org/lkml/b23da846-ff6e-4b73-9691-beb14ceb0fa5@suse.de/T/#u (0.0002)
ubuntu@ip-172-31-10-12:~/cs1380$
```

The page with the highest TF-IDF:

```
The MCP251xFD has a dedicated transceiver standby control function on
the INT0/GPIO0/XSTBY pin, controlled by the XSTBYEN bit in IOCON.
When enabled, the hardware automatically manages the transceiver
standby state: the pin is driven low when the controller is active
and high when it enters Sleep mode.

Enable this feature when the 'microchip,xstbyen' device tree property
is present.

Signed-off-by: Viken Dadhaniya <viken.dadhaniya@oss.qualcomm.com>
---
v2 -> v3:

- Configure xstbyen pin before bringing the controller into normal mode.
- Add a check in mcp251xfd_gpio_request() to ensure that GPIO0 cannot be
  used when xstbyen is enabled.

v2 Link: https://lore.kernel.org/all/20260316131950.859748-3-viken.dadhaniya@oss.qualcomm.com/
---
 .../net/can/spi/mcp251xfd/mcp251xfd-core.c    | 37 +++++++++++++++++++
 drivers/net/can/spi/mcp251xfd/mcp251xfd.h     |  1 +
 2 files changed, 38 insertions(+)

diff --git a/drivers/net/can/spi/mcp251xfd/mcp251xfd-core.c b/drivers/net/can/spi/mcp251xfd/mcp251xfd-core.c
index 9c86df0c2c5..92a86083c896 100644
--- a/drivers/net/can/spi/mcp251xfd/mcp251xfd-core.c
+++ b/drivers/net/can/spi/mcp251xfd/mcp251xfd-core.c
@@ -764,6 +764,31 @@ static void mcp251xfd_chip_stop(struct mcp251xfd_priv *priv,
        mcp251xfd_chip_set_mode(priv, MCP251XFD_REG_CON_MODE_CONFIG);
```