



Source: https://en.wikipedia.org/wiki/Labours_of_Hercules

What's new in BIND 9.20 and what's coming up next?

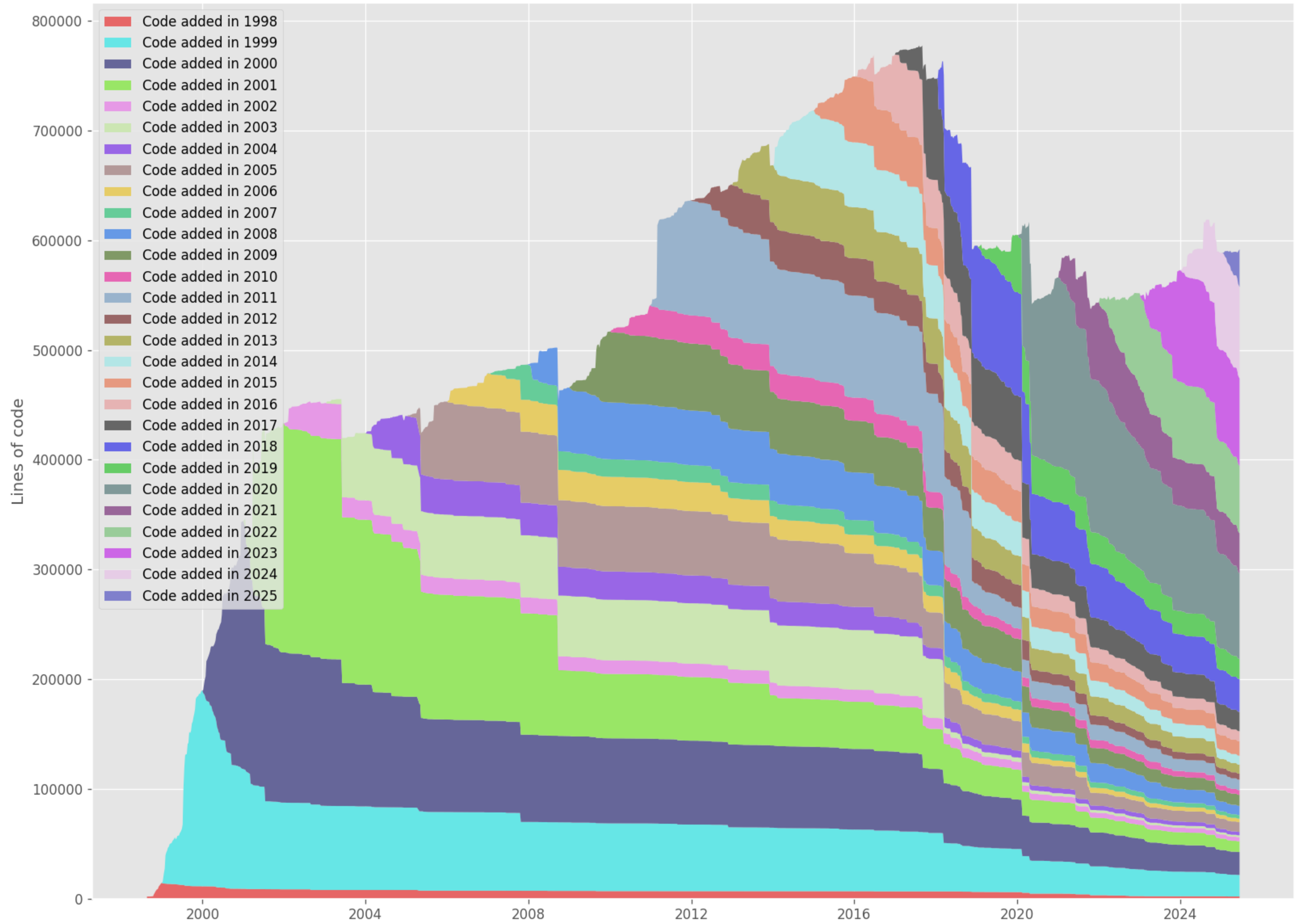
Ondřej Surý, ISC; GRNOG 20.6.2026

BIND 9: History

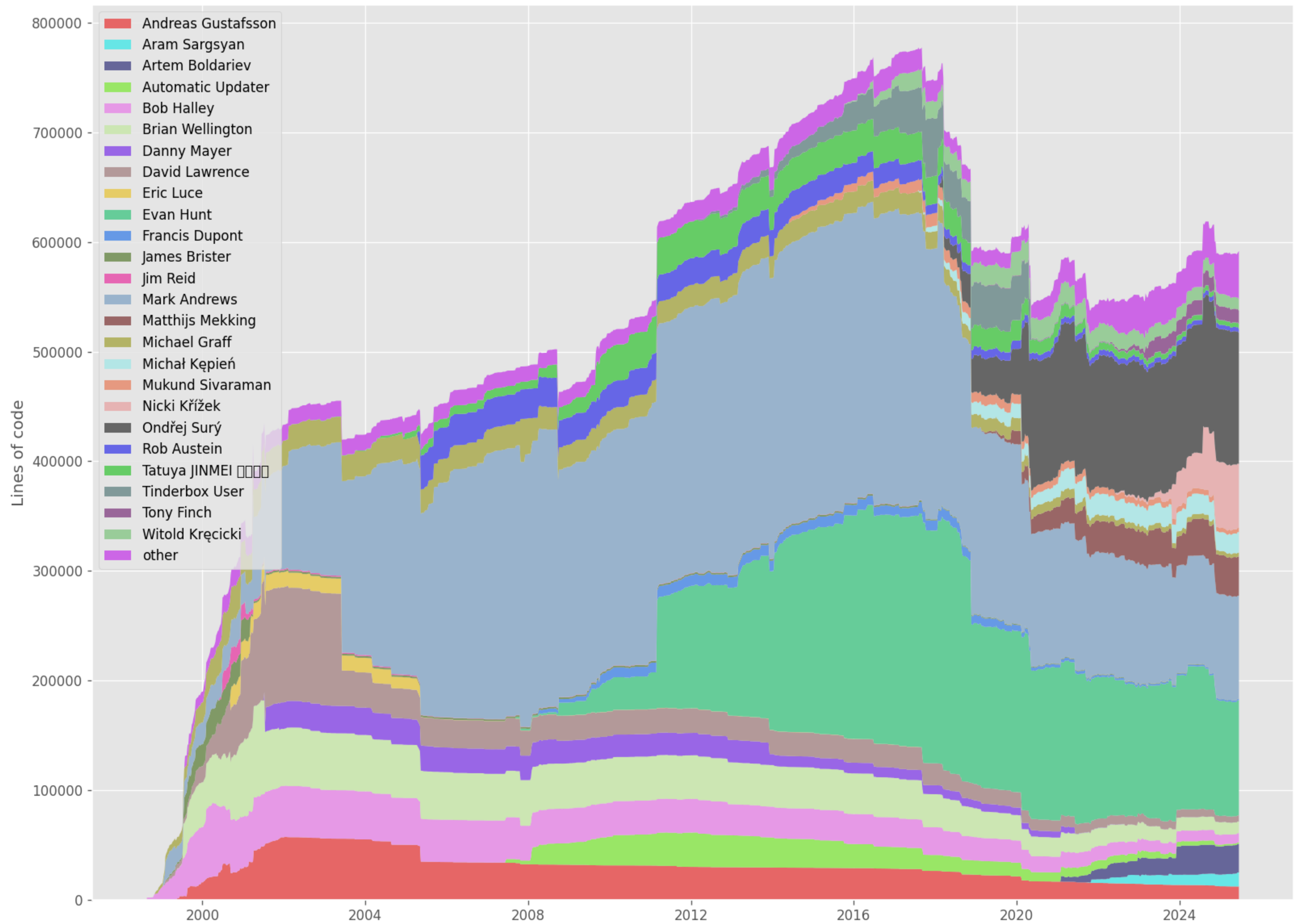
- BIND 9 – development started in 1998
- BIND 9.0.0 – released ~25 years ago!
- BIND 9.12.0 – development model change, more open, more predictable
- BIND 9.20.0 – released in 2024
- Current model:
 - Major release every two years (9.x+2.0)
 - Minor release every month (9.x.y+1)
 - Everything happens in GitLab

<https://www.isc.org/bindhistory/>

BIND 9: History



BIND 9: History



BIND 9.20: what's new?

- DNSSEC configuration updates
 - **dnssec-policy** is now the only option to manage a signed zone
 - **dnssec-keymgr** and **auto-dnssec** options have been removed
- HSM support is configured through dnssec-policy:

```
dnssec-policy {  
    pkcs11-uri <quoted_string>;  
};
```
- PROXYv2 (protocol) is available also for DNS over TCP and DNS over TLS
- Support of Catalog Zone schema version 2

BIND 9.20: what's new?

- Statistics channels are showing incoming zone transfer in progress
 - `http://<ip>:<port>/xml/v3/xfrins`
 - `http://<ip>:<port>/json/v1/xfrins`

ISC Bind 9 Configuration and Statistics

Alternate statistics views: [All](#), [Status](#), [Server](#), [Zones](#), [Network](#), [Memory](#) and [Traffic Size](#)

Server Status

Boot time:	2023-05-30T15:35:36.737Z
Last reconfigured:	2023-05-30T15:35:36.747Z
Current time:	2023-05-30T15:35:53.253Z
Server version:	9.19.14-dev

Incoming Zone Transfers for View _default

Name	Class	Type	Serial	State	Source Address	Destination Address	Transport	Duration	Messages Received	Records Received	Bytes Received	IXFR
example	IN	secondary	1	Initial SOA	10.53.0.3#0	10.53.0.1#5300	TCP	0	0	0	0	No
example2	IN	secondary	5	Finalizing AXFR	10.53.0.3#0	10.53.0.2#24180	TCP	0	1	10	264	No

Incoming Zone Transfers for View _bind

Name	Class	Type	Serial	State	Source Address	Destination Address	Transport	Duration	Messages Received	Records Received	Bytes Received	IXFR
------	-------	------	--------	-------	----------------	---------------------	-----------	----------	-------------------	------------------	----------------	------

BIND 9.20: USDT probes

User Statically-Defined Tracing probes

- Instrumentation of production binaries of BIND 9

User-space program

```
result = foo();  
FIRE_PROBE_FOO_ENDS(result);  
...  
result = bar();  
FIRE_PROBE_BAR_ENDS(result);
```

Kernel module (built/loaded from stap)

```
void fooends(int result) {  
    printf("foo ends with value %d\n", result);  
}  
  
void barends(int result) {  
    printf("bar ends with value %d\n", result);  
}
```

BIND 9.20 adds probes support for rwlock and incoming zone transfer flows

How-To: <https://gitlab.isc.org/isc-projects/bind9/-/wikis/User-space-Probing-in-BIND-9>

BIND 9.20: USDT probes

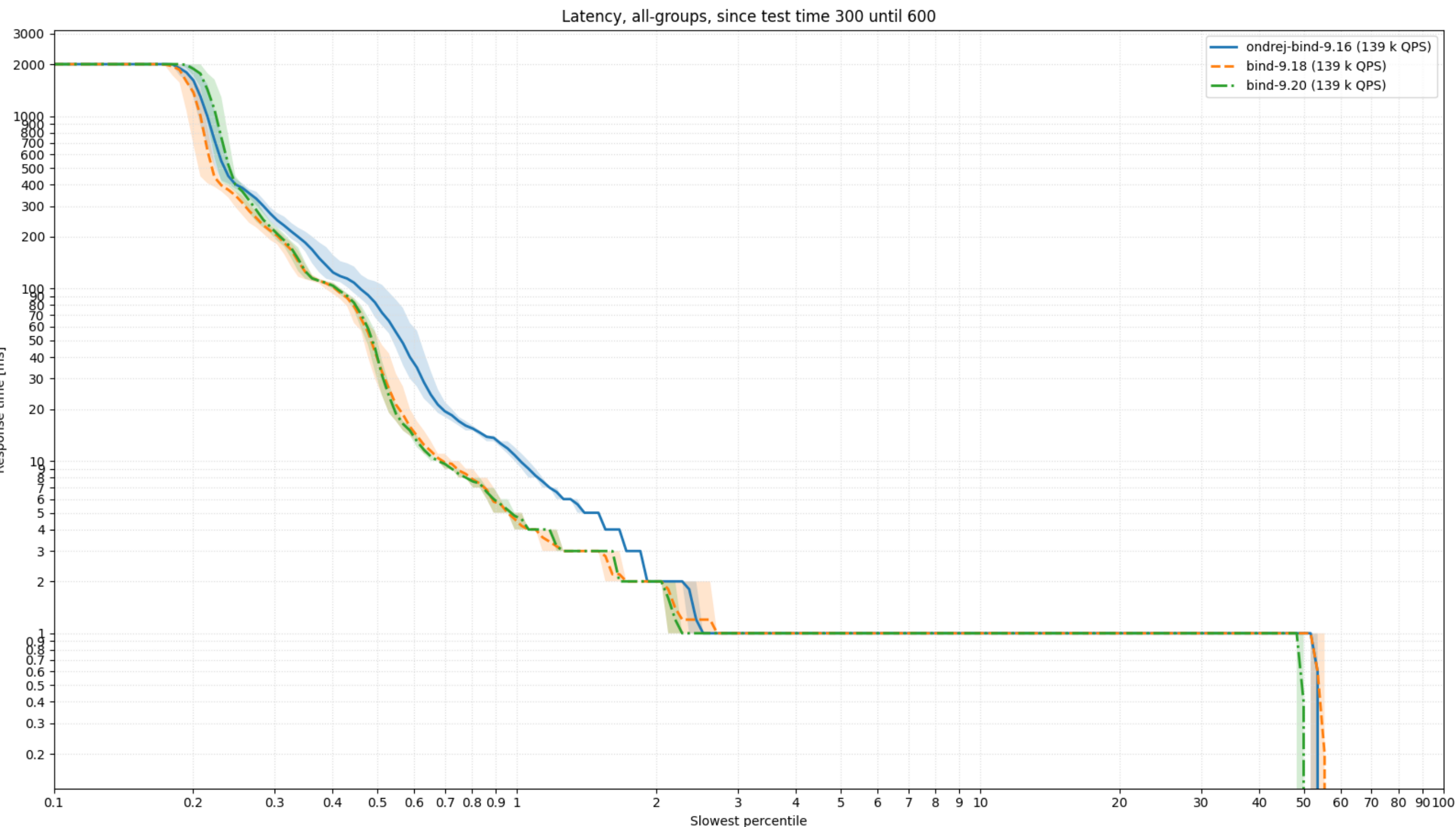
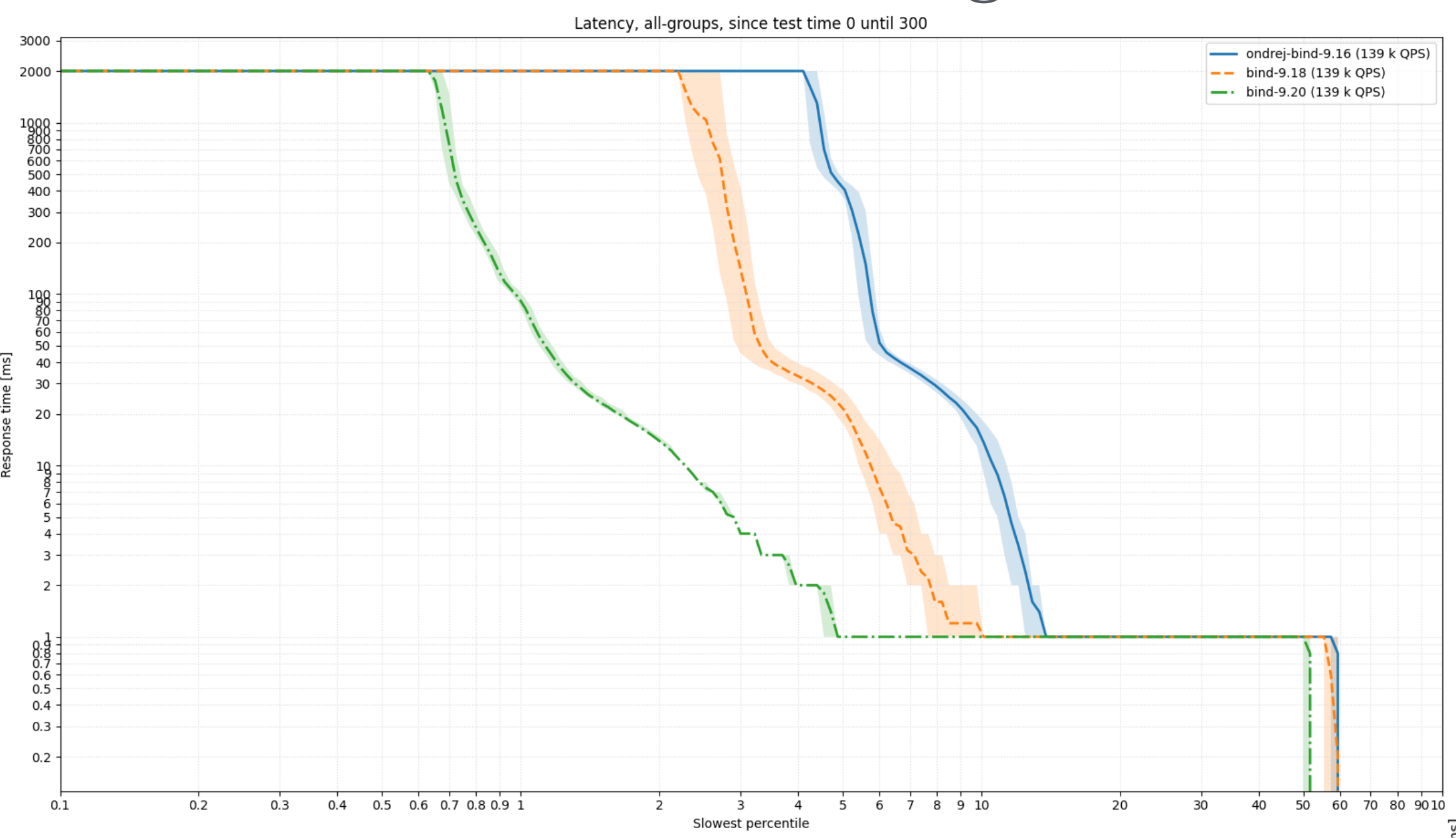
User Statically-Defined Tracing probes

- Example using perf and analyzing incoming transfer

```
# perf buildid-cache --add /usr/local/lib/libdns.so
# perf list | grep xfrin
[... choose the one you're interested in ...]
# perf probe -add="sdt_libdns:xfrin_recv_start"
# perf record -e sdt_libdns:xfrin_recv_start -aR sleep 5
[... trigger zone transfer in named ...]
# perf script
isc-loop-0005 2332966 [002] 940053.314601:
                sdt_libdns:xfrin_recv_start: (7f37d5d28d44)
                                                arg1=139877381898240
                                                arg2=139877381899456
                                                arg3=0
```


BIND 9.20 QP Zone and Cache Database

Includes Networking, Databases and Job Scheduling changes



BIND 9.20: Extended DNS Error

- Response policy zone EDEs 15, 16, 17, 18
 - options {
 - response-policy {
 - zone "example.com." ede none|blocked|censored|filtered|prohibited;
- Since 9.20.6
 - –Unsupported RRSIG algorithm (EDE 1)
 - –Unsupported DNSKEY digest (EDE 2)
 - –Multiple EDEs (up to 3) are supported in the same DNS response
- Since 9.20.8
 - Signature expired (EDE 7)
 - Signature not yet valid (EDE 8)
 - Not authoritative (EDE 20)

BIND 9.20: QPtrie – a new database

- A key-value store which is:
 - Transactional
 - Particularly suited for DNS
- BIND9 implementation of qp-trie is a foundation to a lock-free database:
 - Use Userspace RCU for updates
 - Currently the values stored in the DB still use locking
- Replaced red-black tree (which was using locking)

More info on qp-trie: <https://dotat.at/prog/qp/README.html>

BIND 9.20: libuv event based

- Introduced in 9.16 with a new network manager, initially to handle network events
- In 9.18, replaced all networking
- In 9.20, it replaced the custom-made cooperative scheduling in the whole server.
- Enables to dispatch events in different categories:
 - Very fast, i.e. cached query response
 - Crypto (slow) operations on a dedicated thread-pool
 - Slow and blocking (i.e. IO operations) on another dedicated thread-pool

Simplifies the internal architecture, helps us to focus on what's matter (we're doing DNS, not scheduling), and reduces context-switching as events processing are pinned to threads and the OS takes care of fair thread scheduling.

Future of BIND: new build system

autotools → meson

- Way shorter build time
 - (using hyperfine, Intel Ultra 7 165U machine running Linux 6.14.2)
- Build commands:
 - `meson setup [--prefix=<prefix>] builddir`
`ninja -C builddir`
`meson install -C builddir`
- Requires python3 and ninja

	autotools	meson
configure	6.39s	3.02s
build	29.46s	6.16s

Future of BIND: Zone templates

Simplify the configuration of multiple zones with similar properties

```
template foo {  
    type primary;  
    file "$name.db";  
};
```

```
zone "example1.org" {  
    template foo;  
};
```

```
zone "internal" {  
    template foo;  
    allow-query { 192.168.1.0/24; };  
};
```



```
zone "example1.org" {  
    type primary;  
    file "example1.org.db";  
};
```

```
zone "internal" {  
    type primary;  
    file "internal.db";  
    allow-query { 192.168.1.0/24; };  
};
```


Future of BIND: Zone templates

A template can be built from a template too

```
template foo {  
    type primary;  
    file "$name.db";  
};
```

```
template bar {  
    template foo;  
    allow-update { any; };  
};
```

```
zone "internal" {  
    template bar;  
    allow-query { 192.168.1.0/24; };  
};
```



```
zone "internal" {  
    type primary;  
    file "internal.db";  
    allow-update { any; };  
    allow-query { 192.168.1.0/24; };  
};
```


Future of BIND: Admin API

Current Status

- Need to change a setting from a running server, or add/remove a zone?
 - Update named.conf → rndc reconfig
- Manual process, it stops the server for a little while
 - rndc {add,del,update}zone
 - Carefully read the manpages!
- Need to update an existing zone?
 - Update the zone DB → rndc reload
 - Some changes might be persistent (but not applied in named.conf).
 - Some changes might require manual named.conf changes to persist.
- Manual process, it stops the server potentially for a long while
 - ... BTW, don't forget to increment the SOA number!

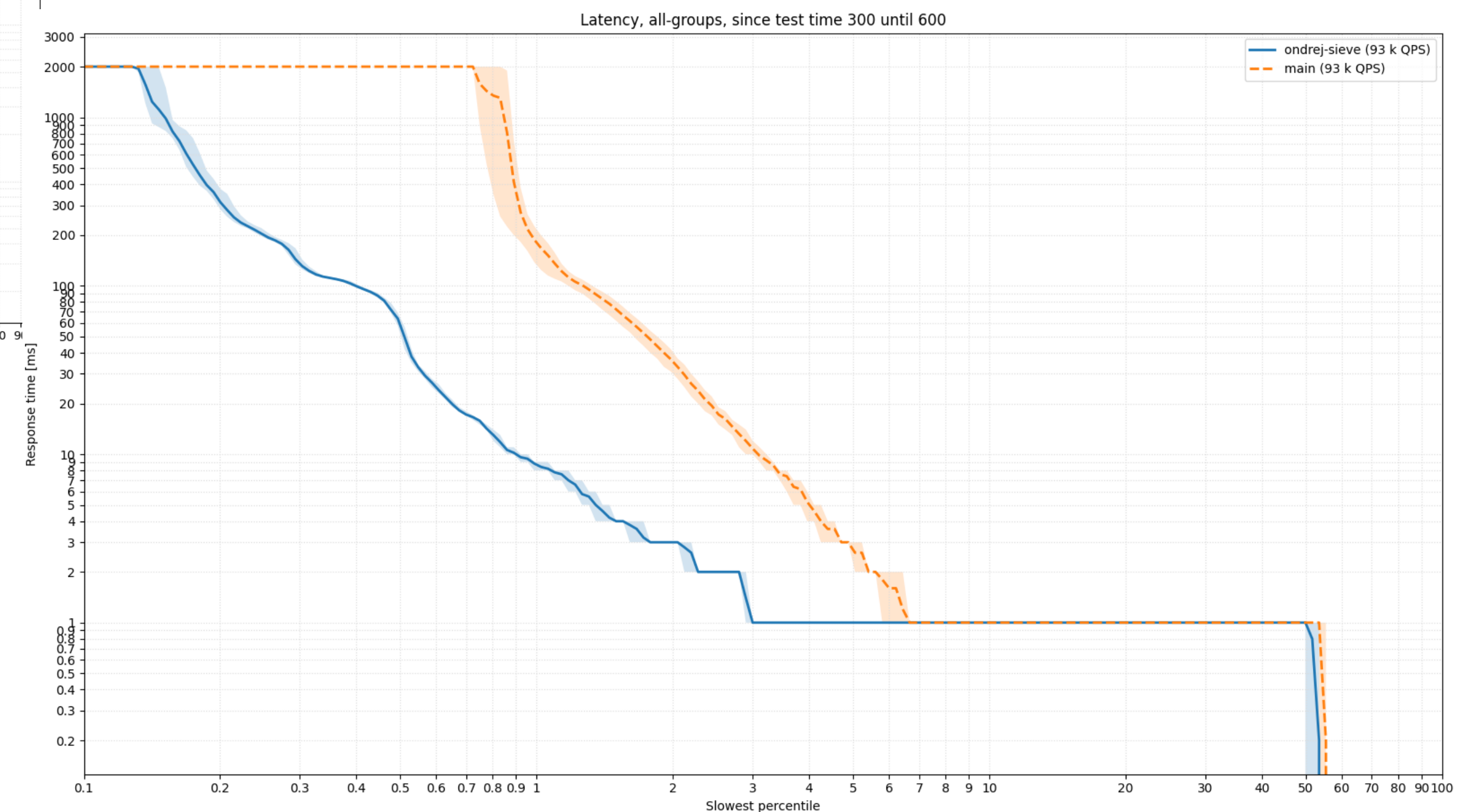
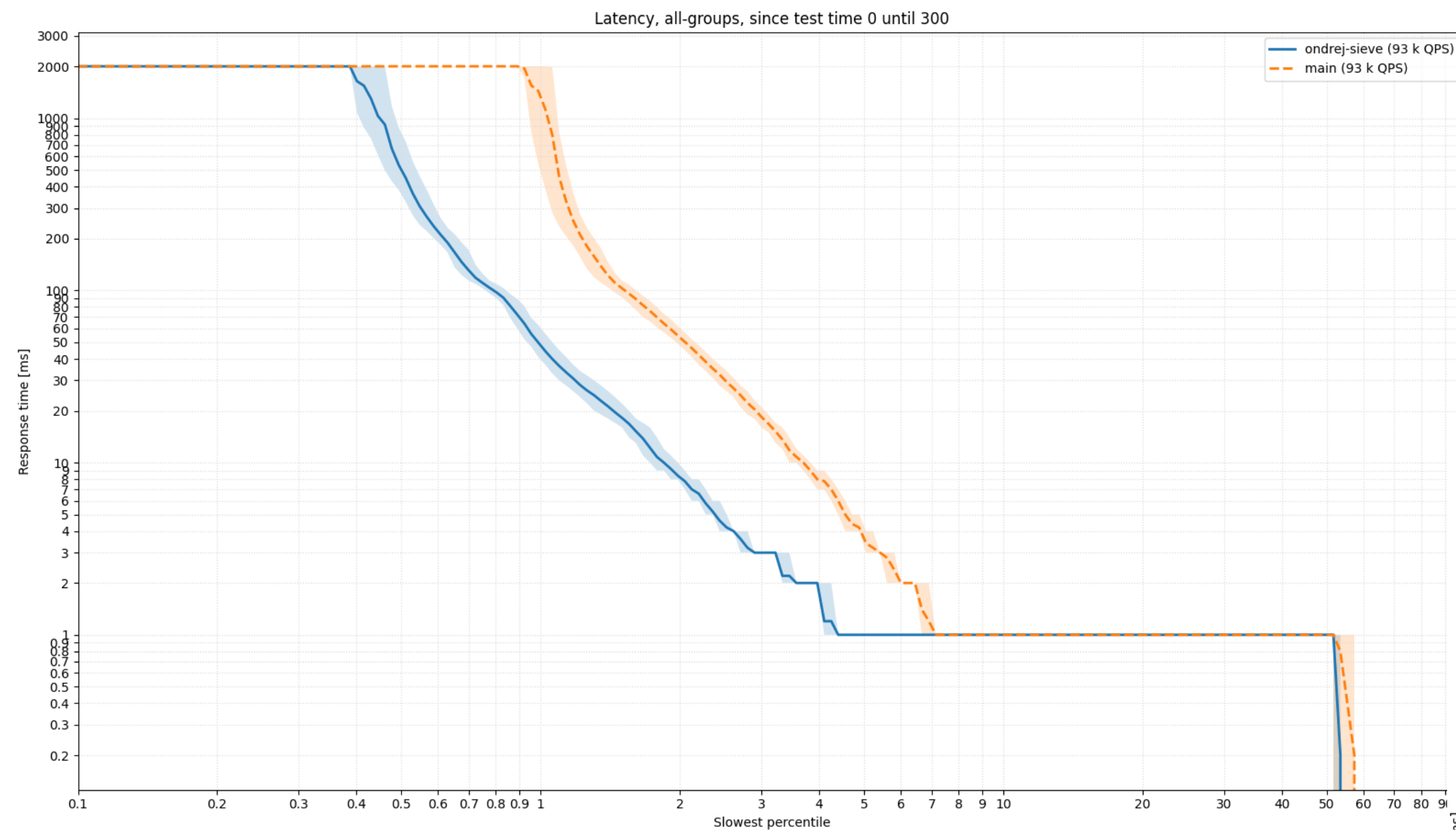
Future of BIND: Admin API

Future goal

- Update setting “X” while running, without traffic disruptions
 - X = { ACL, zone, view, random-config-value-you-name
 - Automatic generation of the “running” named.conf
 - Don’t put the server in an invalid state if a live configuration update is wrong (zone error, wrong ACL, etc.)
 - ...Something goes wrong? Rollback to the previous working state!
 - Make deployment of new configs/ numerous named servers easier
- 
- JSON and REST

Future of BIND: SIEVE instead of LRU

An Eviction Algorithm Simpler than LRU for not only Web Caches



Quite small 128MB cache and heavy traffic!

BIND 9: False Sharing

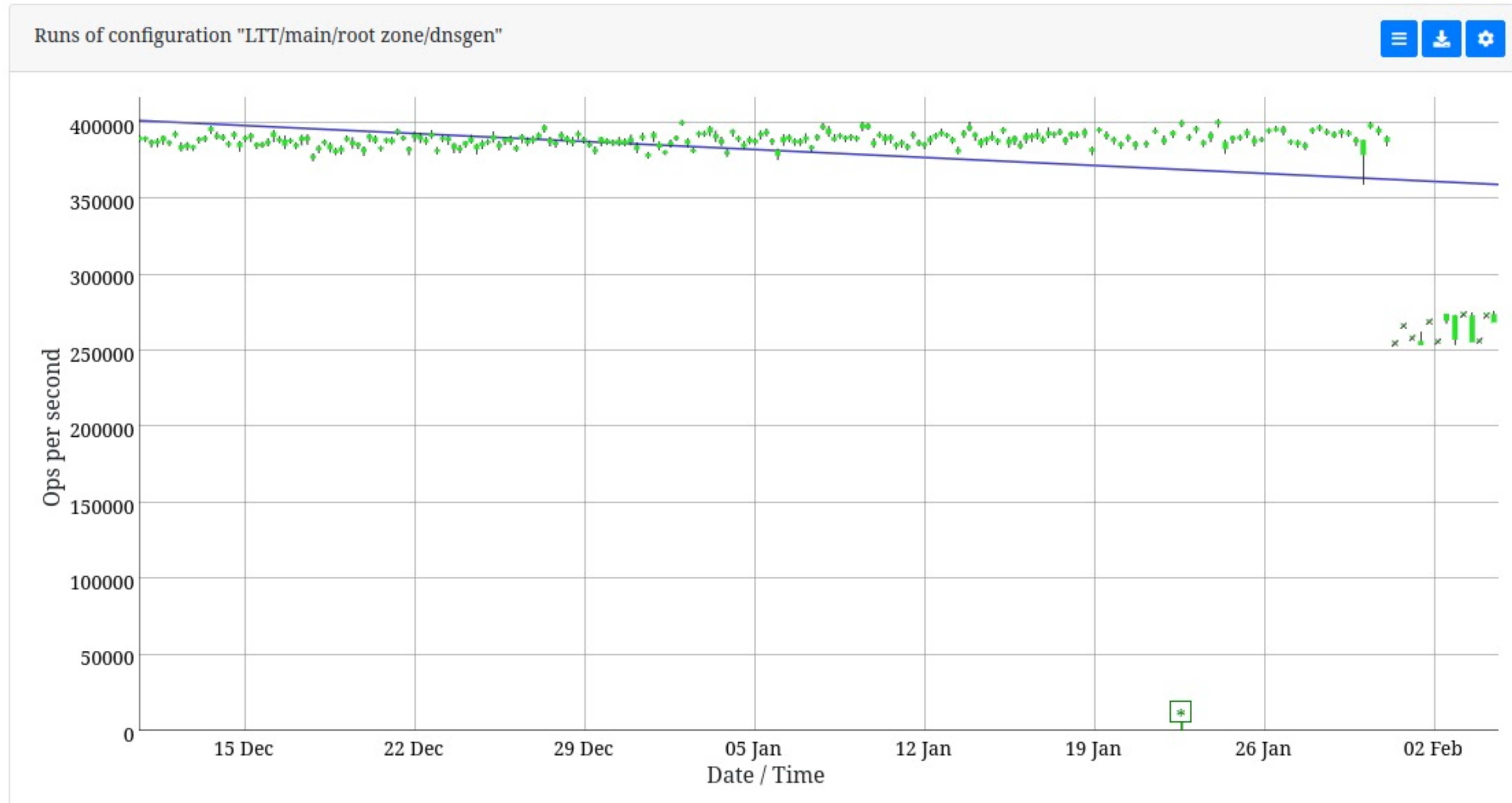
How bad this can be?



In computer science, false sharing is a performance-degrading usage pattern that can arise in systems with distributed, coherent caches at the size of the smallest resource block managed by the caching mechanism.
Source: https://en.wikipedia.org/wiki/False_sharing

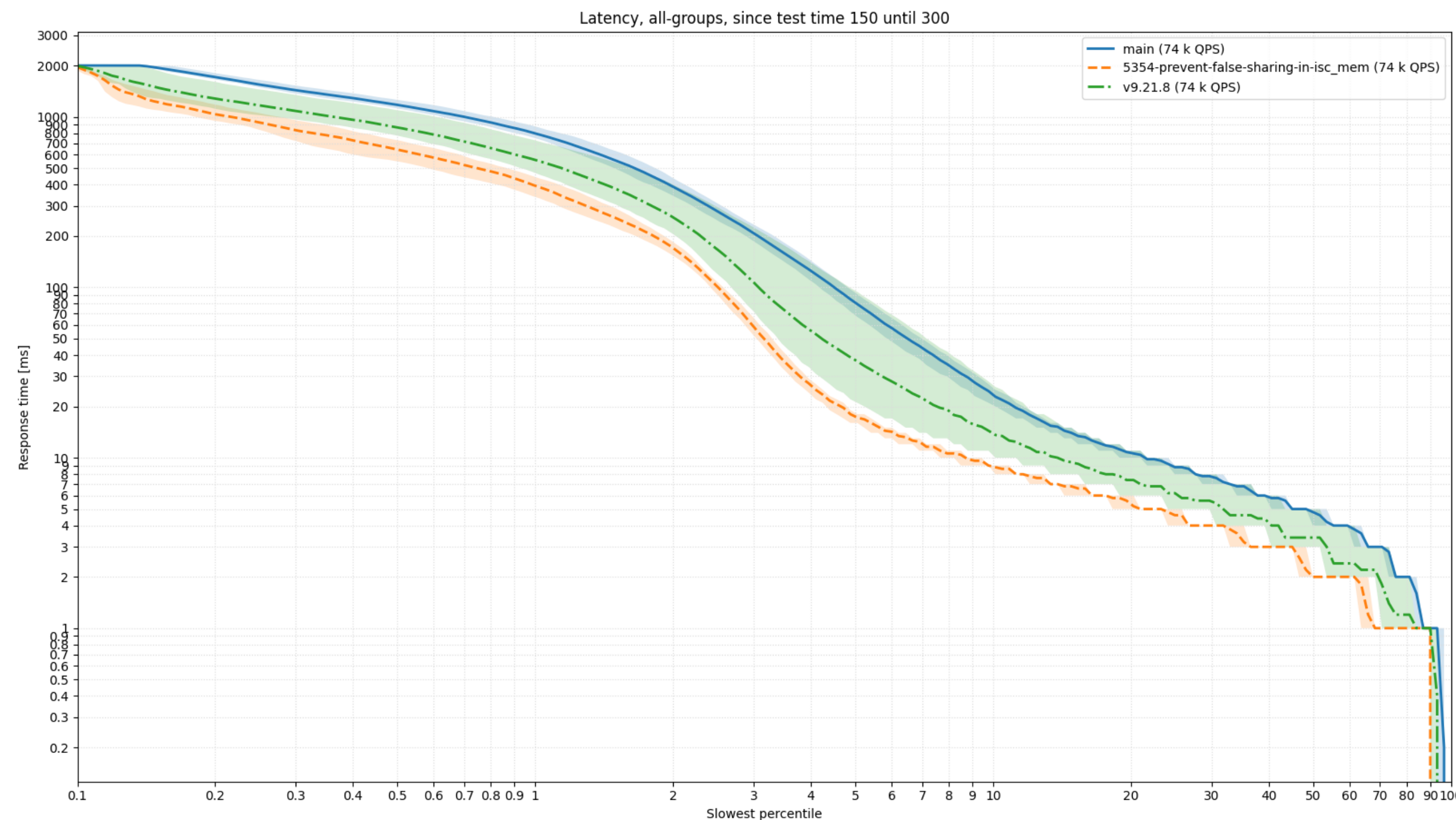
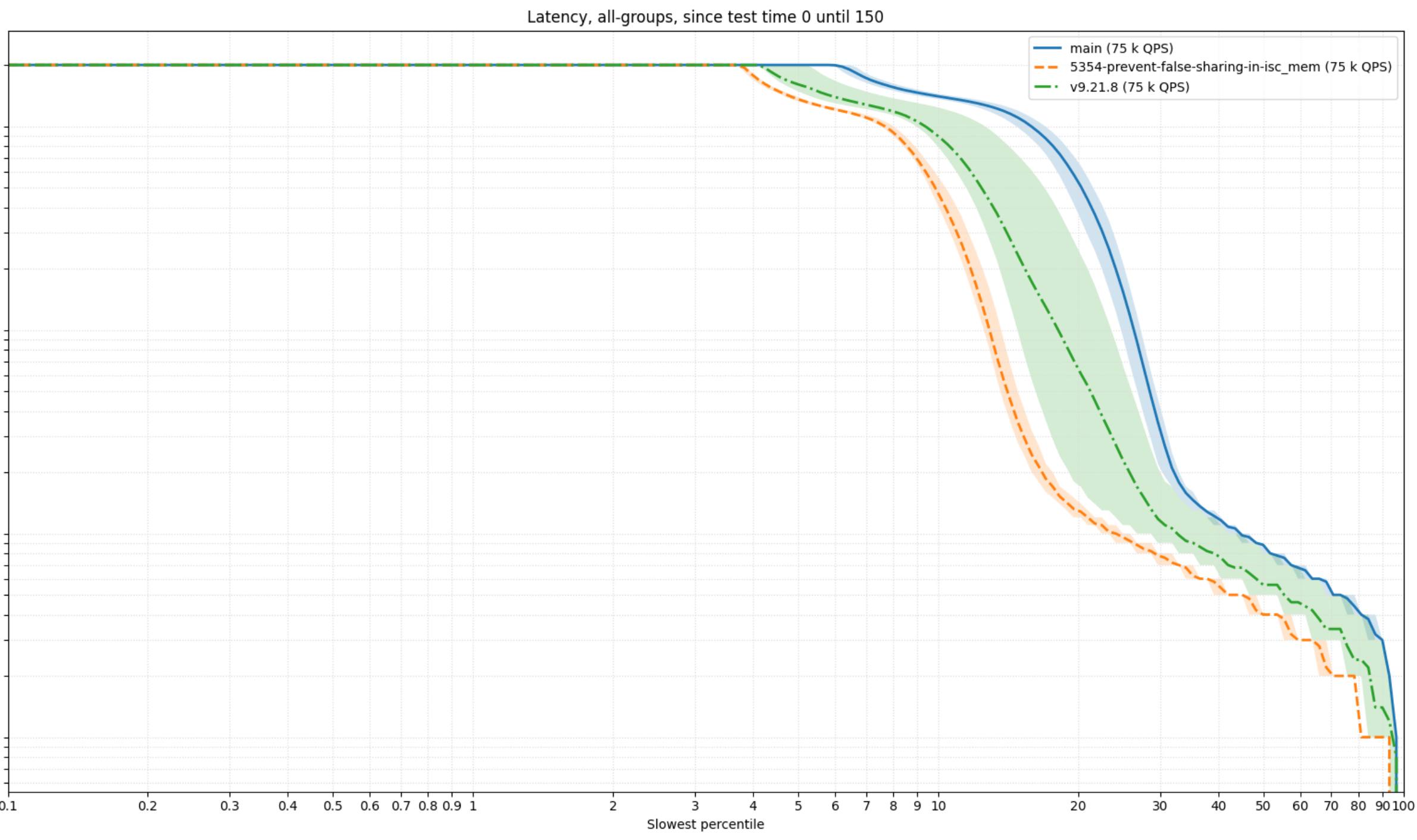
BIND 9: False Sharing

...between locks...



BIND 9: False Sharing

...between atomic counters



Future of BIND: More cool software engineering

- Keep data local to threads (streamline most operations)
- Share only what needs to be shared among threads
- Convert easy and typical cases to RCU API (locking, refcounting)
 - Not everything can be converted – long-lived objects can't use only RCU
- Find the "hot" spots where contention happens
 - This is not only the locking, but also atomic operations

Future of BIND: All your locking are belongs to us!

- Remove locking in QPDB (in order of difficulty)
 - Change the API to use name+type as key (in progress, not so easy)
 - Change the writes to use COW (copy-on-write) mechanism
 - Remove/replace locking on the node buckets
 - TTL-based cleaning (use skiplist instead of heaps, in progress)
 - LRU-based cleaning (quite hard, needs per-thread memory for each node)
- Replace the locking in address database (ADB) and resolver
 - Use lock-free hash-tables (easy)
 - Rewrite the LRU mechanism (hard to harder) <-- oh, maybe not so much with SIEVE



Source: <https://en.wikipedia.org/wiki/Augeas#Mythology>

Thank you!



Ondrej rerouting the rivers Alpheus and Peneus, to clean out the BIND 9 source code. Roman mosaic, 3rd century AD.