Back-asyncmeta status update

Nadezhda Ivanova
Software Engineer @ Symas Corp

LDAPCon2017

Slapd-meta

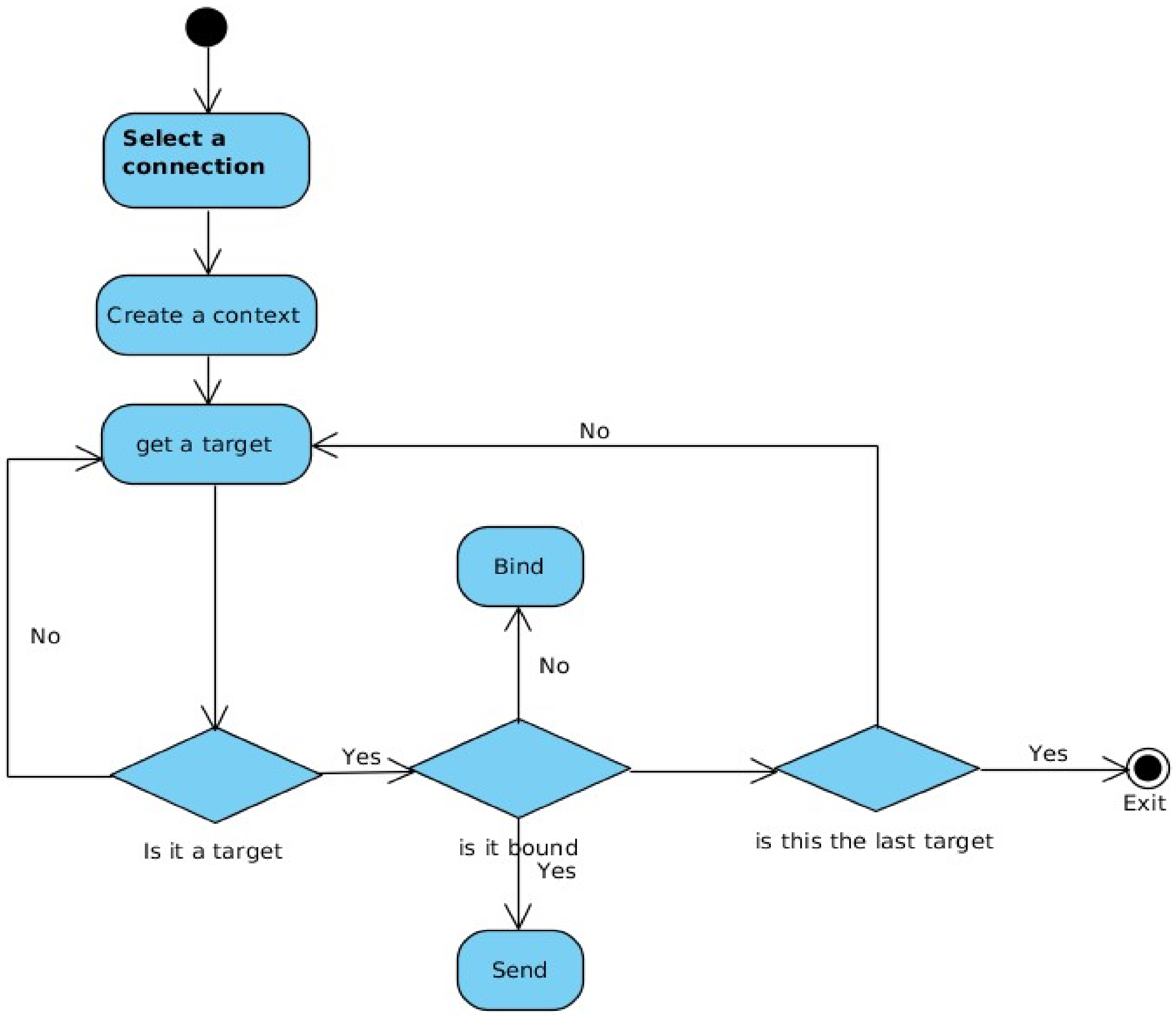
- Ldap-proxy backend, capable of proxying an operation to multiple targets, aggregate the results and present them to the client as if coming from a single server
- Uses one thread per operation to:
  - Find or create a new connection
  - Bind all targets
  - Send to all targets
  - Get entries/results from each target and return them to the client

symas

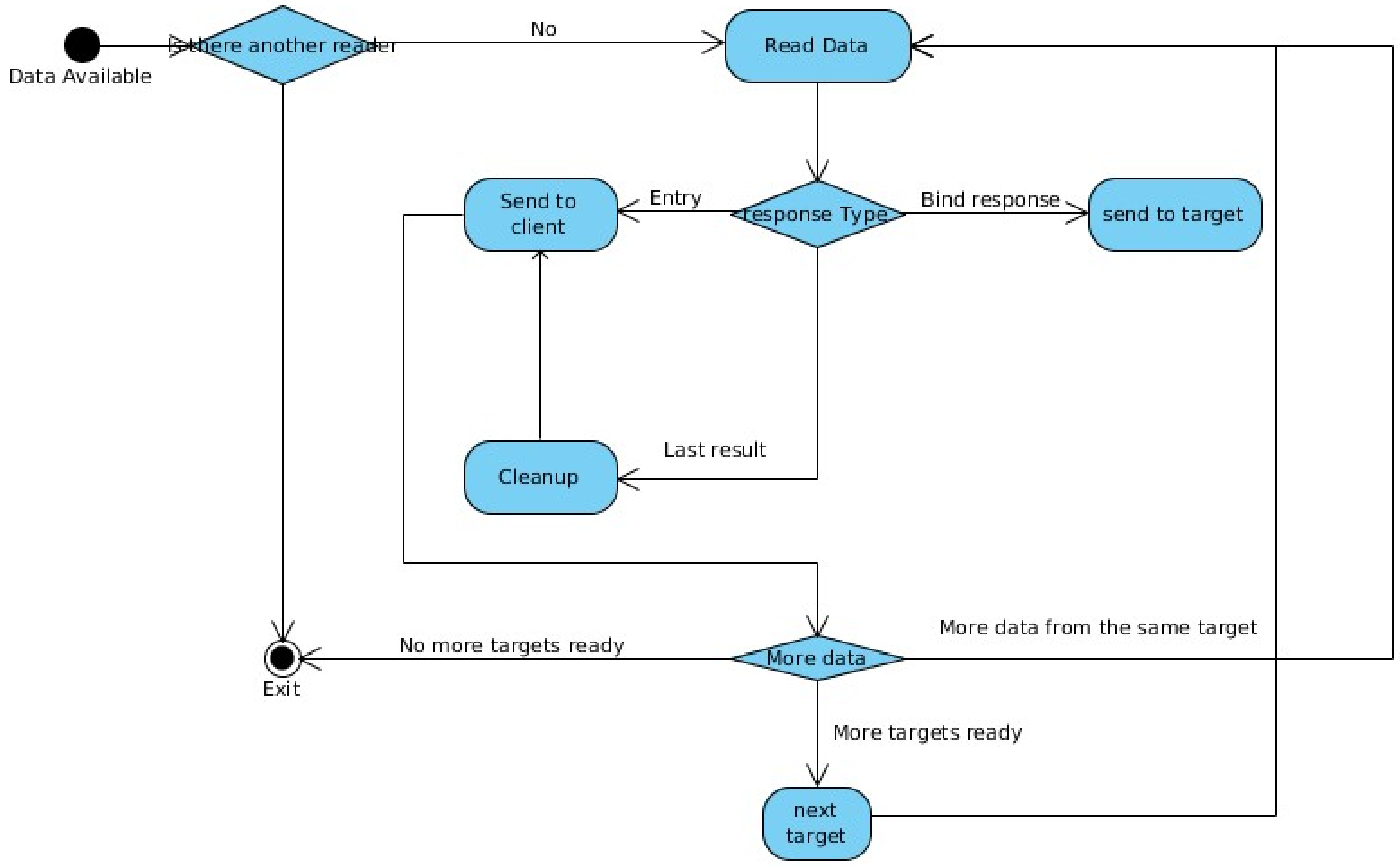
slapd-asyncmeta

- Ldap-proxy backend, capable of proxying an operation to multiple targets, aggregate the results and present them to the client as if coming from a single server
- Processing of operations is asynchronous and therefore executed by more than one thread:
  - One thread to encode, send the request to the target server and add it to the pending queue
  - Another to read the response from the target and relay it back to the client

Sender


Select a connection
Create a context
get a target
No
Bind
No
No
Yes
Yes
Exit
Is it a target
is it bound
Yes
is this the last target
Send

Reader


Data Available
Is there another reader
No
Read Data
Response Type
Entry
Bind response
Send to client
send to target
Last result
Cleanup
No more targets ready
More data from the same target
More data
Exit
More targets ready
next target

What's changed since LDAPCon2015

- Substantial improvements in stability and memory management
  - Fixed memory leaks, multiple functional issues and crashes, tested at maximum system loads and different network conditions
- Some changes to configuration and functionality
  - No rewrite engine – only suffixmassage on DN available
  - Network-timeout now configurable in milliseconds. Sets the network timeout value after which poll(2)/select(2) following a connect(2) returns in case of no activity when sending an operation to the remote target.
- ITS#8734

symas

Lessons

Multi-threaded programming

Theory

Practice

- Adding an asynchronous back-end to a front-end that does not expect it – bad idea
- Using your data in a way it's not meant to be used – another bad idea
- Expect the unexpected (use cases)
- As you approach the speed of light, weird things happen to time

Why use asyncmeta?

- Slightly decreased or equal throughput in ideal network conditions
- Limited functionality – no rewrite engine
- Well...

Setup

- A single slapd server, with 3 databases configured
  - An asyncmeta database with one target
  - A meta database with one target – the target server is the same as asyncmeta
  - A local mdb database
- A slamd, running 2 jobs simultaneously, no traffic restrictions:
  - 5 clients performing a one-level search on the mdb database
  - 5 clients performing the identical search on a meta or asyncmeta database

No network delay

Meta and mdb

Avg per second	Avg duration	# of entries	Result codes
1782.993	56.080	1.000	Success(100%)
2548.804	39.231	1.000	Success(100%)

Asyncmeta and mdb

Avg per second	Avg duration	# of entries	Result codes
1532.683	65.233	1.000	Success(100%)
21418.232	4.714	1.000	Success(100%)

300 ms network latency

Meta and mdb

Avg per second	Avg duration	# of entries	Result codes
57.280	1743.354	1.000	Success (100%)
7127.022	14.029	1.000	Success (100%)

Asyncmeta and mdb

Avg per second	Avg duration	# of entries	Result codes
314.120	318.258	1.000	Success (100%)
24476.402	4.084	1.000	Success (100%)

50% packet loss

Meta and mdb

Avg per second	Avg duration	# of entries	Result codes
24.139	3983.972	1.000	Success (100%)
20.815	4786.157	1.000	Success (100%)

Asyncmeta and mdb

Avg per second	Avg duration	# of entries	Result codes
59.633	1667.900	1.000	Success (61.302%) admin limit exceeded (38.698%)
16817.889	5.944	1.000	Success (100%)

When to use slapd-meta

- You need the rewrite engine – suffixmassage is not enough
- When you are secure in the quality of your network
- When proxying is your server's “life mission”.
- When you need to use any of slapd's existing overlays

When to consider use slapd-asyncmeta

- When suffix-massage will do
- When your slapd server is expected to handle local traffic, apart from proxying
- When fluctuations in network performance are expected
- When you do not need existing overlays, or are willing to test and fix potential issues

symas

Caveats

- No guarantee it will work with any existing overlays
- In custom modules, do not use o_tmpmem allocator for o_extra data
- Make sure the callbacks are dynamically allocated.