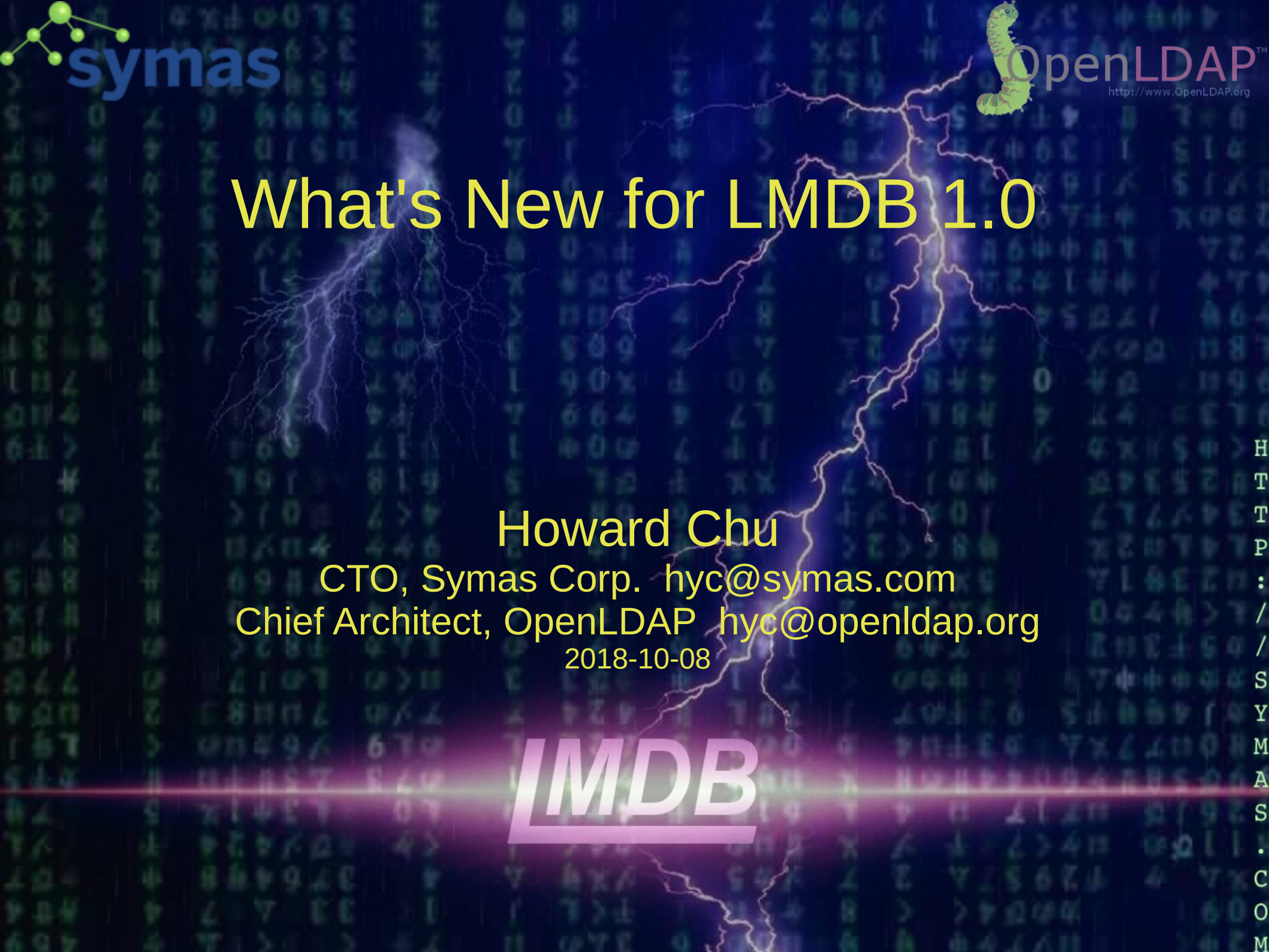

symas
OpenLDAP
http://www.OpenLDAP.org
What's New for LMDB 1.0
Howard Chu
CTO, Symas Corp. hyc@symas.com
Chief Architect, OpenLDAP hyc@openldap.org
2018-10-08
LMDB
HTTP://SYMAS.COM

A Word About Symas

- Founded 1999
- Founders from Enterprise Software world
  - *platinum* Technology (Locus Computing)
  - IBM
- Howard joined OpenLDAP in 1999
  - One of the Core Team members
  - Appointed Chief Architect January 2007
- No debt, no VC investments: self-funded

Intro

- Howard Chu
  - Founder and CTO Symas Corp.
  - Developing Free/Open Source software since 1980s
    - GNU compiler toolchain, e.g. "gmake -j", etc.
    - Many other projects...
  - Worked for NASA/JPL, wrote software for Space Shuttle, etc.

Topics

(1) LMDB so far (0.9)

(2) Coming in 1.0

(1) LDMB so far (0.9)

- Initially set out to replace BerkeleyDB in OpenLDAP alone, but now supported across multiple fields:
  - OS level - rpm, git, SASL, Kerberos, Samba
  - Cryptocurrency blockchains
  - AI / deep learning / machine learning
  - High frequency trading
  - High performance computing

(1) LDMB so far (0.9)

- Released in 2011
  - Still the world's smallest, fastest and most reliable transactional embedded storage engine
- Now widely adopted
  - Dozens of wrappers for other languages (34 at last count)
  - Deployed as primary storage backend for countless open source and proprietary projects

(2) Coming in 1.0

- Support for 64bit DBs on 32bit builds (VL32)
- Page level encryption and/or checksums
- Incremental backup
- Long key support (up to 2GB keys)
- Semi-synchronous writes

(2) Coming in 1.0

- Headerless overflow pages
- Restructured freelist
- Raw partition support
- 2-phase commit

(2) Coming in 1.0

- On-disk format change
  - Most of these features require a change to the LMDB page formats
  - Some of these features have been working for years, but we didn't want to release them separately and have to deal with multiple DB migrations

(2) VL32

- Optional support for larger-than-32bit databases on 32bit builds
  - Instead of using a single mmap for the entire DB, maps in chunks of >= 16 pages at a time
  - Requires some user-level caching, slower than native 32bit support
    - Must track referenced pages per transaction and within the entire environment
- Allows DBs to be used interchangeably between 32bit and 64bit machines
  - If using semaphores instead of shared mutexes, DBs can be shared by 32bit and 64bit processes on the same machine

(2) Page encryption

- (Never thought we'd ever do this)
  - Requires app-level page caching
    - to keep a decrypted copy of a page being accessed
    - But we already have this, due to VL32 support
- Ciphers in user-supplied callbacks
  - there will be no built in encryption or checksum algorithms
- Needs space for per-page checksums

(2) Incremental Backup

- Adds transaction ID to page header
  - Allows simple backup of all pages newer than a specified txnID
  - Incremental backups can easily be overlaid on top of a snapshot from the specified txnID

(2) Long Keys

- Support for storing keys on overflow pages
  - Key length up to 2^31 -1 bytes
  - Values still limited to 2^32 -1 bytes

(2) SemiSync Writes

- Allow asynchronous writes without risk of corruption
  - Uses 2 new temporary meta pages for recording the semisync commits
  - Use explicit mdb_env_sync() to persist temporary meta pages onto main meta pages
  - In case of crash, un-synced commits will be lost, but main meta pages will be intact
  - Will incur additional DB growth while un-synced

(2) Headerless OVF

- Currently the first page in a span of overflow pages always has a standard page header (16 bytes on 64bit build)
- So, storing a page-size value consumes two pages, because it can't all fit on the first page
- Relevant header info will be moved into the leaf node instead
  - page number, transaction ID, checksum

(2) Freelist

- Currently the freelist is stored as a single record per transaction
  - performance suffers when the free space is fragmented
- Instead, will use DUPSORT|DUPFIXED to let LMDB store each freelist entry in arbitrary pages
  - Total space requirement will be about the same
  - But will be immune to fragmentation issues

(2) Raw partitions

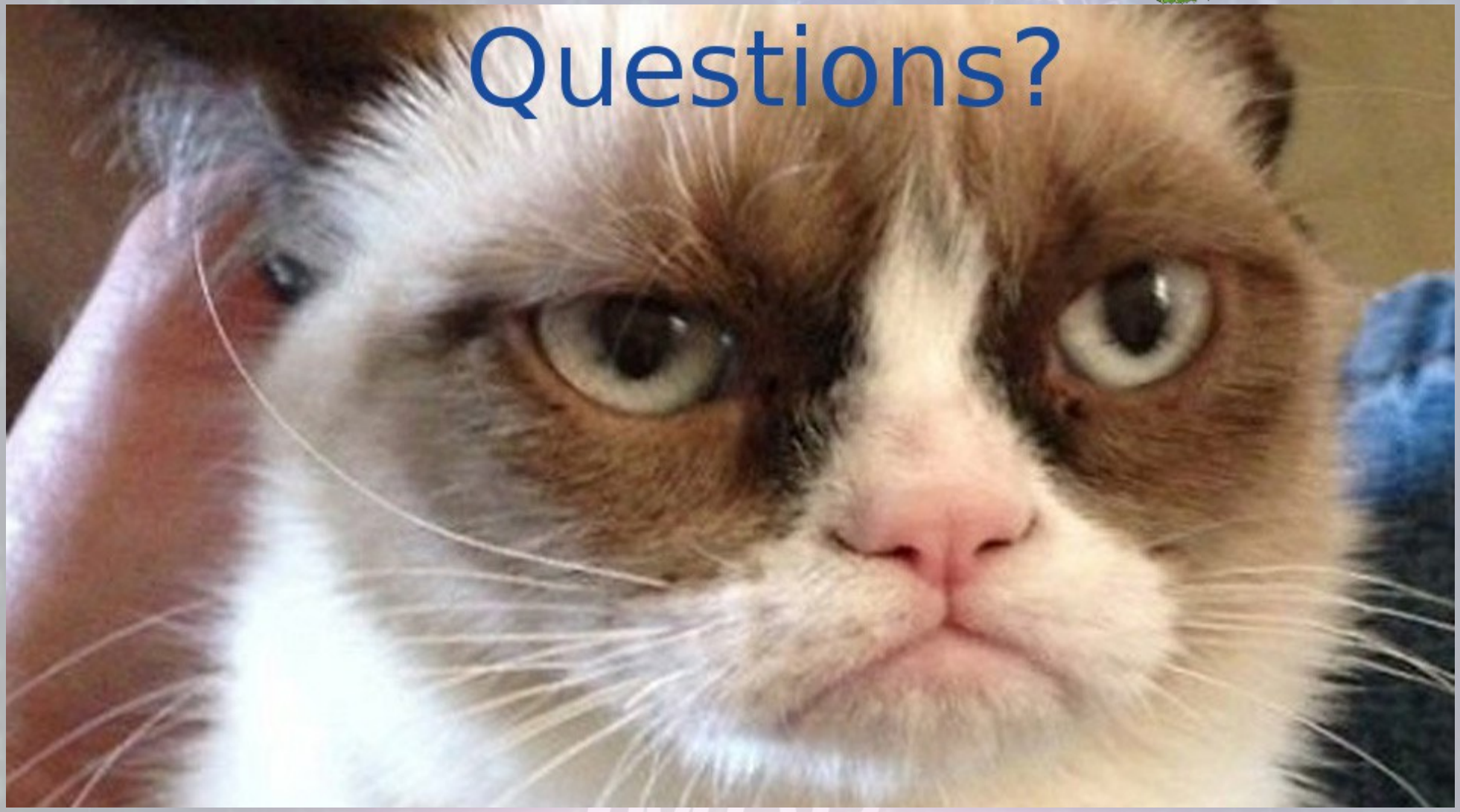
- Allow storing LMDB data directly on a raw block or character device
  - Avoids all filesystem metadata overhead
    - Bulk load on raw device is 2x faster than on ext4fs with journaling disabled
    - No other performance impact
  - Makes maxsize discussion totally moot
  - Allows exploiting special features of NVDIMMs

(2) 2-Phase Commit

- Add an mdb_txn_prepare() API
- Enables support for distributed transactions
  - For use with slapd back-ldap / back-meta / etc.

symas

OpenLDAP™
http://www.OpenLDAP.org



LMDB

HTTP://SYMAS.COM