Stealth Databases: Ensuring User-Controlled Queries in Untrusted Cloud Environments

Josef Spillner, Martin Beck, Alexander Schill, Thomas Michael Bohnert
<josef.spillner@zhaw.ch>

Service Prototyping Lab (blog.zhaw.ch/icclab)
Zurich University of Applied Sciences, Switzerland

December 9, 2015 | 8th IEEE/ACM UCC, Limassol, Cyprus
Intro: Data Management in the Cloud

«We did this 7 years ago.» (cloud databases/data management, DBLP)

«We did this 40 years ago.» (networked databases)

Unsolved challenges:
• Maintaining availability and confidentiality of data
• User-controlled multi-criteria optimisation
• Applicability to cluster, cloud and streaming environments
Data Service Quality Concerns

↑ Availability
~99.9999% available.

↑ Reliability
100% available, 100% correct.

↑ Confidentiality
not readable for third parties

↑ Scalability, Resilience, ...

↓ Volume, Price, ...

Intrusion

Risks

Dropbox Sync Error Causes Data Loss For Customers

Loss

Downtime

Dropbox is experiencing issues.

Leak

Bankruptcy

Price Jump

New pricing and channel changes:
On November 19, 2013, Amazon announced an increase in price from $99/year to $999/year. The move sparked an intense reaction from users at the company's forum, even though existing users were grandfathered into the original pricing plan. Reaction from bloggers was particularly critical.

AWS users fret over downtime ahead of Amazon's massive EC2 reboot

We are saying Good Bye...
Stealth Idea

Stealth layer: Coverable cloud service evolution

s: cloud services (PaaS, IaaS)
Stealth Concepts Overview

Stealth Computing: Well-protected processing of data in the cloud [NetSys'15, BlackSeaCom'15]

Concepts
- Data coding and distribution
- Data processing
- User preferences and quality constraints

Approach: operation-aware data multi-coding, distribution and processing
- in one pipeline for data records and data streams over multiple independent services evolving over time

- full compute: turing-complete: functions, containers, VMs
- restricted compute: expressions: SQL, CQL, arithmetics and search
- no compute: storage, transmission
Concept: Multi-Coded Distributed Data

Coding

Steganography (S) → S+C → S+C+D → D+S

Dispersion (D) → D+C → S+C+E+D → D+S+E

Confidentiality/Privacy

Encryption (E) → E+S → E+C → E+C+D

Goals:
availability → executable
dispersion confidentiality/privacy → executable
encryption capacity → executable
compression

Executable Multi-Coding
= «Stealth Data»
Concept: Multi-Coded Distributed Data

Distribution: find optimal placement according to desired availability with minimum redundancy overhead

Placement algorithm PICav+: Improved Precise, Iterative, Complement-based availability calculation [UCC'14]

\[ C_n \subseteq C_h \in \mathcal{P}(C) = \{\{T_1\}, \{T_1, T_2\}, ..., \{T_1, ..., T_n\}\} \]
Concept: Stealth Data Processing

Local processing in each location
• resource provider view: random data blocks
• application view: map-carry-reduce access to full results
Concept: Type-Ops-Dependent Coding

Strings
• exact string search
• redundant fragments: fuzzy string search
• fragment selection: heuristics

Integer numbers
• exact arithmetics
• sampling: approximate arithmetics
• fragment selection: heuristics

Floating point numbers
• fragment selection: varying-precision processing

Multimedia
• fragment selection: interpolated processing
Concept: Query Optimisation

Syntax: `SELECT ... OPTIMIZE FOR <goal(s)>`

Preferences

- performance
  - apply fragment selection & sampling
- precision
  - request all fragments (even beyond default)
- reliability
  - query fully replicated values & compare
- energy efficiency
  - upon sorting: apply sweetspot CPU frequency
- power
  - upon sorting: apply lowest CPU frequency
StealthDB System Overview

Resources
• local storage (RAM, files) + compute
• remote storage+compute services

(Processable) dispersion
• replication, hashring, erasure, bitsplit

(Processable) encryption
• homomorphic, order-preserving, searchable, fuzzy

Preferences
• optimises queries for performance, reliability, energy efficiency, precision

Features
• data records and streams
• per-column distribution and migration control
• map-carry-reduce operators
StealthDB Architecture

Coding: Dispersion
• erasure | bitsplitting | hashring replication

Coding: Encryption
• homomorphic + order-preserving + searchable + diffuse

Features:
• datasets & streams
• per-column distribution
• migration control
• map-carry-reduce operations
• user requirements optimisation
• dynamic deployment of processing code
StealthDB Software

Laboratory approach: live demo - recomputable results / reproducibility

josef@rumba:/repos/space-universe/dispersedalgorithms/db$ ./stealthdb
~~ StealthDB >master >Wed May 20 16:14:37 2015 +0200 ~~
Type HELP; to get started.
Using database 'stealthdb'.
Storing all data and performing all procedures on ['mem://localhost'] with ['replication'].
>>> HELP;
StealthDB Quickhelp
HELP [<topic>]
SHOW DATABASES|TABLES
CREATE TABLE <table> [([<column> <column-type>, ...])
DESCRIBE <table>
DROP TABLE [IF EXISTS] <table>
CREATE DATABASE <database>
USE DATABASE <database>
DROP DATABASE <database>
[EXPLAIN ANALYZE] SELECT [DISTINCT] */<column>/<aggregate>(*<column>)/<predicate>, ... [FROM <table>]
[WHERE <column> LIKE/=/* <value> ] [ORDER BY <column> [ASC|DESC]] [OPTIMIZE FOR <goal>] [FOREVER]
INSERT INTO <table> ( <column>, ...) VALUES ( <value>, ... )
DELETE FROM <table>
USE CLOUDS <cloud> [AND <cloud>...][WITH <distribution>]
ALTER TABLE <table> [ALTER COLUMN <column>] USE CLOUDS ...
MODE <mode>
>>>
Functionality Evaluation: Stealth Apps

Embedding StealthDB
- low-level: Python methods
- high-level: SQL parser method
- transparent: as a network proxy/service

So far
- 3 stealth web applications
- 1 stealth IoT streaming prototype
Performance Evaluation

Diverse configurations - Raspberry Pi cluster, Amazon EC2, localhost...

Note: quantitative considerations to be generalised at some point
Summary & Future Work

Achievements
• tiny but powerful data processing prototype for mixed setups
  • clouds, clusters (RPC), in-memory, files, ...
• user-friendly query optimisations to ensure SLAs can be met
• novel design paradigm for inherent quality in cloud applications

Future work
• effort/benefit comparison with other reliability and security techniques
e.g. cloud-native applications
• marketplaces full of stealthy applications!

Thanks to ad-hoc collaboration partners

\[\text{DIE} \quad \text{TI} \quad \text{MOOSLAB} \quad \text{unibz} \quad \text{TECHNISCHE UNIVERSITÄT DRESDEN} \]
References


