gem version 2.1.1 build passing coverage 88%

Introduction

The Tozny End-to-End Encrypted Database (E3DB) is a storage platform with powerful sharing and consent management features. Read more on our blog.

E3DB provides a familiar JSON-based NoSQL-style API for reading, writing, and querying data stored securely in the cloud.

Installation

Add this line to your application's Gemfile:

```
gem 'e3db'
```

And then execute:

```
$ bundle
```

Or install it yourself as:

```
$ gem install e3db
```

At runtime, you will need the `libsodium` cryptography library required by the native RbNaCl Ruby library. On most platforms a package is available by default:

```
$ brew install libsodium            (Mac OS X)
$ apt-get install libsodium-dev     (Ubuntu)
```

For more information including libsodium installation instructions for Windows, see the libsodium web site.

Windows Users: Make sure to download a recent "MSVC" build. Once downloaded, find the most recent `libsodium.dll` inside the ZIP file, rename it to `sodium.dll` and copy it to C:\usr\local\lib. You can also copy it to your \Windows\System32 directory.

Registering a client

Register an account with InnoVault to get started. From the Admin Console you can create clients directly (and grab their credentials from the console) or create registration tokens to dynamically create clients with `E3DB::Client.register()`. Clients registered from within the console will automatically back their credentials up to your account. Clients created dynamically via the SDK can *optionally* back their credentials up to your account.

For a more complete walkthrough, see `/examples/registration.rb`.

Without Credential Backup

```
token = '...'
client_name = '...'

public_key, private_key = E3DB::Client.generate_keypair
client_info = E3DB::Client.register(token, client_name, public_key)
```

The object returned from the server contains the client's UUID, API key, and API secret (as well as echos back the public key passed during registration). It's your responsibility to store this information locally as it *will not be recoverable* without credential backup.

With Credential Backup

```
token = '...'
client_name = '...'

public_key, private_key = E3DB::Client.generate_keypair
client_info = E3DB::Client.register(token, client_name, public_key,
private_key, true)
```

The private key must be passed to the registration handler when backing up credentials as it is used to cryptographically sign the encrypted backup file stored on the server. The private key never leaves the system, and the stored credentials will only be accessible to the newly-registered client itself or the account with which it is registered.

Loading configuration and creating a client

Use the `E3DB::Config.default` class method to load the default client configuration, and pass it to the `E3DB::Client` constructor:

```
require 'e3db'
config = E3DB::Config.default
client = E3DB::Client.new(config)
```

Using profiles to manage multiple configurations

The E3DB Command-Line Interface allows you to register and manage multiple keys and credentials using *profiles*. To register a new client under a different profile:

```
$ e3db register --profile=development developers@mycompany.com
```

You can then use `E3DB::Config.load_profile` to load a specific profile inside your Ruby application:

```
config = E3DB::Config.load_profile('development')
client = E3DB::Client.new(config)
```

Writing a record

To write new records to the database, call the `E3DB::Client#write` method with a string describing the type of data to be written, along with a hash containing the fields of the record. `E3DB::Client#write` returns the newly created record.

```
record = client.write('contact', {
  :first_name => 'Jon',
  :last_name => 'Snow',
  :phone => '555-555-1212'
})
printf("Wrote record %s\n", record.meta.record_id)
```

Querying Records

E3DB supports many options for querying records based on the fields stored in record metadata. Refer to the API documentation for the complete set of options that can be passed to `E3DB::Client#query`.

For example, to list all records of type `contact` and print a simple report containing names and phone numbers:

```
client.query(type: 'contact') do |record|
  fullname = record.data[:first_name] + ' ' + record.data[:last_name]
  printf("%-40s %s\n", fullname, record.data[:phone])
end
```

In this example, the `E3DB::Client#query` method takes a block that will execute for each record that matches the query. Records will be streamed efficiently from the server in batches, allowing processing of large data sets without consuming excessive memory.

In some cases, it is more convenient to load all results into memory for processing. To achieve this, instead of passing a block to `E3DB::Client#query`, you can call `Enumerable` methods on the query result, including `Enumerable#to_a` to convert the results to an array.

For example:

```
results = client.query(type: 'contact').to_a
printf("There were %d results.\n", results.length)
results.each do |record|
  puts record
end
```

More examples

See the simple example code for runnable detailed examples.

Development

Before running tests, register an account with InnoVault, and generate a client token.

After checking out the repo, run `bin/setup` to install dependencies. Next, set two environment variables:

- API_URL - E3DB host to run tests against.
- REGISTRATION_TOKEN - A token obtained from InnoVault that can be used to register E3DB clients.

Run `rake spec` to run the tests. You can also run `bin/console` for an interactive prompt that will allow you to experiment.

To install this gem onto your local machine, run `bundle exec rake install`. To release a new version, update the version number in `version.rb`, and then run `bundle exec rake release`, which will create a git tag for the version, push git commits and tags, and push the `.gem` file to rubygems.org.

Documentation

General E3DB documentation is on our web site.

Comprehensive documentation for the SDK can be found online via RubyDoc.info.

Contributing

Bug reports and pull requests are welcome on GitHub at https://github.com/tozny/e3db-ruby.