Intel® Cluster Checker Analysis API

Generated by Doxygen 1.6.1

Mon Aug 3 16:34:09 2015

Contents

Chapter 1

Todo List

Class clck::Diagnosis A new type field will be added to the Fault class. This class will be removed.

Member clck::Layer::collect() Placeholder - not implemented

Class clck::Layer::Config Additional configuration options are likely to be added in the future, as needed.

Class clck::Sign A new type field will be added to the Fault class. This class will be removed.

Chapter 2

Class Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

Class Documentation

4.1 clck::Layer::Config Struct Reference

Layer configuration options.

```
#include <clck.h>
```

Public Types

- enum { **DATABASE**, **NODELIST**, **INTERSECTION**, **UNION** }

Public Member Functions

- Config ()
- Config (std::shared_ptr< Database > db, const std::vector< std::string > &extension_mods, const std::vector< std::string > &kb_mods, const std::string &extension_path="", const std::string &kb_path="")

Public Attributes

- std::shared_ptr< Database > db
- time_t expiration = 0
- std::vector< std::string > extension_mods
- std::string extension_path
- std::vector< std::string > kb_mods
- std::string kb_path
- std::string language

- std::vector< Node > nodes
- enum clck::Layer::Config:: { ... } node_source
- time_t now = time(NULL)
- std::vector< ConfigParam > config_params

4.1.1 Detailed Description

Layer configuration options.

Todo

Additional configuration options are likely to be added in the future, as needed.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 clck::Layer::Config::Config ()

Default Config constructor

4.1.2.2 clck::Layer::Config::Config (std::shared_ptr< Database > *db*, const std::vector< std::string > & *extension_mods*, const std::vector< std::string > & *kb_mods*, const std::string & *extension_path* = "", const std::string & *kb_path* = "")

Config constructor

Parameters:

db Instance of class derived from Database

extension_mods List of connector extensions to be loaded

kb_mods List of knowledge base files to be loaded

extension_path Absolute path to the connector extension directory

kb_path Absolute path to the knowledge base directory

4.1.3 Member Data Documentation

4.1.3.1 std::vector<ConfigParam> clck::Layer::Config::config_params

Data analysis configuration parameters

4.1.3.2 std::shared_ptr<Database> clck::Layer::Config::db

Instance of class derived from Database

4.1.3.3 time_t clck::Layer::Config::expiration = 0

Maximum allowable age of data, relative to the value of now. Data older than now minus expiration will be ignored. A value of 0 means to use all data, i.e., there is no expiration.

4.1.3.4 std::vector<std::string> clck::Layer::Config::extension_mods

List of connector extensions to be loaded

4.1.3.5 std::string clck::Layer::Config::extension_path

Absolute path to the connector extension directory

4.1.3.6 std::vector<std::string> clck::Layer::Config::kb_mods

List of knowledge base files to be loaded

4.1.3.7 std::string clck::Layer::Config::kb_path

Absolute path to the knowledge base directory

4.1.3.8 std::string clck::Layer::Config::language

String representing the message catalog language

4.1.3.9 enum { ... } clck::Layer::Config::node_source

Select the source for the list of nodes. Options are to use the database (exclusively), the input nodes vector (exclusively), or the intersection or union of the two.

4.1.3.10 std::vector<Node> clck::Layer::Config::nodes

List of nodes to be analyzed (see node_source).

4.1.3.11 time_t clck::Layer::Config::now = time(NULL)

Reference time to be used as the current time for analysis

4.2 clck::Layer::ConfigParam Struct Reference

Data analysis configuration parameter.

```
#include <clck.h>
```

Public Attributes

- std::string module
- std::string key
- std::vector< std::string > values

4.2.1 Detailed Description

Data analysis configuration parameter.

4.2.2 Member Data Documentation

4.2.2.1 std::string clck::Layer::ConfigParam::key

Parameter name

4.2.2.2 std::string clck::Layer::ConfigParam::module

Module name in the knowledge base

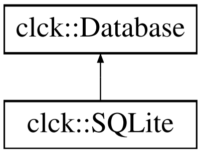
4.2.2.3 std::vector<std::string> clck::Layer::ConfigParam::values

Parameter value(s)

4.3 clck::Database Class Reference

Base class for database configuration.

`#include <clck.h>`Inheritance diagram for clck::Database::



Public Member Functions

- virtual ∼Database ()=0

4.3.1 Detailed Description

Base class for database configuration.

4.3.2 Constructor & Destructor Documentation

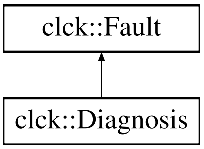
4.3.2.1 virtual clck::Database::∼Database () `[pure virtual]`

Database destructor

4.4 clck::Diagnosis Class Reference

A diagnosis is the root cause of an issue. Diagnosis is derived from the Fault class.

`#include <clck.h>`Inheritance diagram for clck::Diagnosis::



4.4.1 Detailed Description

A diagnosis is the root cause of an issue. Diagnosis is derived from the Fault class.

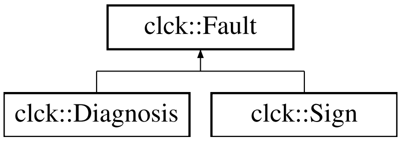
Todo

A new type field will be added to the Fault class. This class will be removed.

4.5 clck::Fault Class Reference

A fault is the basic analysis unit. A fault is either a sign (i.e., observation) or a diagnosis (i.e., root cause).

`#include <clck.h>`Inheritance diagram for clck::Fault::



Public Attributes

- int confidence = 0
- std::string id
- std::string msg
- std::vector< std::string > nodes
- std::string remedy
- int severity = 0
- bool suppressed = false
- std::set< int > rowid

4.5.1 Detailed Description

A fault is the basic analysis unit. A fault is either a sign (i.e., observation) or a diagnosis (i.e., root cause).

4.5.2 Member Data Documentation

4.5.2.1 int clck::Fault::confidence = 0

Confidence percentage (0 - 100)

4.5.2.2 std::string clck::Fault::id

Message catalog id

4.5.2.3 std::string clck::Fault::msg

Expanded message string

4.5.2.4 std::vector<std::string> clck::Fault::nodes

List of nodes

4.5.2.5 std::string clck::Fault::remedy

Expanded remedy string

4.5.2.6 std::set<int> clck::Fault::rowid

DB rows that provide the raw data upon which the sign/diagnosis is based

4.5.2.7 int clck::Fault::severity = 0

Severity percentage (0 - 100)

4.5.2.8 bool clck::Fault::suppressed = false

True if the diagnosis / sign is suppressed, false otherwise

4.6 clck::Layer::Filter Struct Reference

Filter for the list of faults returned by get_faults().

```
#include <clck.h>
```

Public Attributes

- int confidence = 0
- std::vector< std::string > ids
- std::vector< std::string > nodes
- int severity = 0
- std::bitset< 2 > state = CLCK_FAULT_STATE_DIAGNOSED | CLCK_FAULT_STATE_OBSERVED
- std::bitset< 2 > suppressed = CLCK_FAULT_SUPPRESSED_FALSE
- std::bitset< 2 > type = CLCK_FAULT_TYPE_DIAGNOSIS | CLCK_FAULT_TYPE_SIGN

4.6.1 Detailed Description

Filter for the list of faults returned by get_faults().

4.6.2 Member Data Documentation

4.6.2.1 int clck::Layer::Filter::confidence = 0

Select faults with a greater than or equal to confidence value.

4.6.2.2 std::vector<std::string> clck::Layer::Filter::ids

Select faults corresponding to at least one of the ids. If empty, does not filter on id.

4.6.2.3 std::vector<std::string> clck::Layer::Filter::nodes

Selects faults corresponding to at least one of the nodes. If empty, does not filter on node.

4.6.2.4 int clck::Layer::Filter::severity = 0

Select faults with a greater than or equal to severity value.

4.6.2.5 std::bitset<2> clck::Layer::Filter::state = CLCK_FAULT_STATE_DIAGNOSED | CLCK_FAULT_STATE_OBSERVED

Select faults with a matching state. Only applies to signs, not diagnoses.

4.6.2.6 std::bitset<2> clck::Layer::Filter::suppressed = CLCK_FAULT_SUPPRESSED_FALSE

Select faults with a matching suppression value.

4.6.2.7 std::bitset<2> clck::Layer::Filter::type = CLCK_FAULT_TYPE_DIAGNOSIS | CLCK_FAULT_TYPE_SIGN

Select faults with a matching type.

4.7 clck::Layer Class Reference

The presentation layer.

```
#include <clck.h>
```

Classes

- struct Config

  Layer configuration options.

- struct ConfigParam

  Data analysis configuration parameter.

- struct Filter

  Filter for the list of faults returned by get_faults().

- struct Message

  Internal Layer messages for the caller to handle.

- struct Sorting

  Sort order for the list of faults returned by get_faults().

- struct Suppression

  Suppress faults matching the specified values.

Public Member Functions

- Layer (const Config &config)
- ~Layer ()
- bool analyze (const std::vector< Suppression > &suppressions)
- bool collect ()
- std::vector< std::shared_ptr< Fault > > get_faults (const Filter &filter, const std::vector< Sorting > &sorting)
- std::vector< Message > get_messages ()
- std::vector< Node > get_nodes ()
- int get_version_number ()
- bool progress (unsigned long &remaining, unsigned long &completed)

Public Attributes

- std::condition_variable message

4.7.1 Detailed Description

The presentation layer.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 clck::Layer::Layer (const Config & *config*)

Layer constructor

4.7.2.2 clck::Layer::∼Layer ()

Layer destructor

4.7.3 Member Function Documentation

4.7.3.1 bool clck::Layer::analyze (const std::vector< Suppression > & *suppressions*)

Start the analysis. Note: behavior is undefined if invoked more than once per Layer instance.

Returns:

> True if the analysis was completed successfully, false otherwise. Note: this does NOT reflect the state of the cluster.

4.7.3.2 bool clck::Layer::collect () `[inline]`

Collect data.

Todo

> Placeholder - not implemented

Returns:

> True if the data collection was completed successfully, false otherwise.

4.7.3.3 std::vector<std::shared_ptr<Fault> > clck::Layer::get_faults (const Filter & *filter*, const std::vector< Sorting > & *sorting*)

Returns a list of faults. May be called multiple times. While analyze() is active, presumably in another thread, the behavior is undefined. Use progress() to determine current analysis status. If called before analyze(), returns an empty list.

Parameters:

filter Return only faults that match the filter

sorting List of sorting criteria. The first element is the primary sorting criterion, the second element the secondary sorting criterion, etc. If empty, the faults are returned unsorted.

Returns:

A list of faults

4.7.3.4 std::vector<Message> clck::Layer::get_messages ()

Returns a list of messages generated internal to Layer for the caller to handle.

Returns:

A list of messages

4.7.3.5 std::vector<Node> clck::Layer::get_nodes ()

Returns the list of nodes to be analyzed.

Returns:

A list of nodes.

4.7.3.6 int clck::Layer::get_version_number ()

Returns the version of the Layer API.

Returns:

The version number. Version X.Y.Z is represented as $(X*1000000 + Y*1000 + Z)$.

4.7.3.7 bool clck::Layer::progress (unsigned long & *remaining*, unsigned long & *completed*)

While analyze() is active, presumably in another thread, returns the number of rules remaining to be fired and the number of rules already run. If called before analyze(), both values will be 0.

Parameters:

remaining The number of rules remaining to be fired. Not guaranteed to be monotonic. Returned by reference.

completed The number of rules that have been fired. Will be monotonic. Returned by reference.

Returns:

False if analyze() has not yet been called, true otherwise.

4.7.4 Member Data Documentation

4.7.4.1 std::condition_variable clck::Layer::message

Notify when a new internal Layer message is available

4.8 clck::Layer::Message Struct Reference

Internal Layer messages for the caller to handle.

```
#include <clck.h>
```

Public Attributes

- int level
- std::string msg

4.8.1 Detailed Description

Internal Layer messages for the caller to handle.

4.8.2 Member Data Documentation

4.8.2.1 int clck::Layer::Message::level

Message level (priority). Inherits logging levels from syslog.

4.8.2.2 std::string clck::Layer::Message::msg

Message string

4.9 clck::Node Class Reference

The node container.

```
#include <clck.h>
```

Public Types

- enum role_t {

  ROLE_BOOT, **ROLE_COMPUTE**, **ROLE_ENHANCED**, **ROLE_EXTERNAL**,

  ROLE_HEAD, **ROLE_JOB_SCHEDULE**, **ROLE_LOGIN**, **ROLE_NETWORK_ADDRESS**,

  ROLE_STORAGE }

Public Attributes

- std::string subcluster
- std::string name
- std::vector< role_t > roles

4.9.1 Detailed Description

The node container.

4.9.2 Member Enumeration Documentation

4.9.2.1 enum clck::Node::role_t

Roles than a node can fulfill.

4.9.3 Member Data Documentation

4.9.3.1 std::string clck::Node::name

An unique node identifier, i.e., the hostname.

4.9.3.2 std::vector<role_t> clck::Node::roles

A list of roles that the node fulfills.

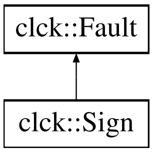
4.9.3.3 std::string clck::Node::subcluster

A subcluster to which the node belongs.

4.10 clck::Sign Class Reference

A sign is an observation of an issue. Sign is derived from the Fault class.

`#include <clck.h>`Inheritance diagram for clck::Sign::



Public Types

- enum { **DIAGNOSED**, **OBSERVED** }

Public Attributes

- enum clck::Sign:: { ... } state

4.10.1 Detailed Description

A sign is an observation of an issue. Sign is derived from the Fault class.

Todo

A new type field will be added to the Fault class. This class will be removed.

4.10.2 Member Data Documentation

4.10.2.1 enum { ... } clck::Sign::state

Either diagnosed (meaning used to make a diagnosis) or observed (undiagnosed).

4.11 clck::Layer::Sorting Struct Reference

Sort order for the list of faults returned by get_faults().

```
#include <clck.h>
```

Public Types

- enum { **CONFIDENCE**, **ID**, **NODE**, **SEVERITY** }

Public Member Functions

- Sorting (bool ascending, decltype(field) field)

Public Attributes

- bool ascending = true
- enum clck::Layer::Sorting:: { ... } field

4.11.1 Detailed Description

Sort order for the list of faults returned by get_faults().

4.11.2 Constructor & Destructor Documentation

4.11.2.1 clck::Layer::Sorting::Sorting (bool *ascending*, decltype(field) *field*)

Sorting constructor

4.11.3 Member Data Documentation

4.11.3.1 bool clck::Layer::Sorting::ascending = true

If true, sort in ascending order, otherwise sort in descending order.

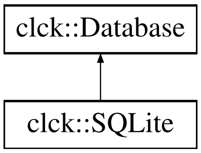
4.11.3.2 enum { ... } clck::Layer::Sorting::field

Fault field to sort on

4.12 clck::SQLite Class Reference

SQLite configuration. Derived from Database.

`#include <clck.h>`Inheritance diagram for clck::SQLite::



Public Member Functions

- SQLite (const std::string &db_file)
- ∼SQLite ()

Public Attributes

- std::string db_file

4.12.1 Detailed Description

SQLite configuration. Derived from Database.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 clck::SQLite::SQLite (const std::string & *db_file*)

SQLite constructor

4.12.2.2 clck::SQLite::∼SQLite ()

SQLite destructor

4.12.3 Member Data Documentation

4.12.3.1 std::string clck::SQLite::db_file

Absolute path to the database file

4.13 clck::Layer::Suppression Struct Reference

Suppress faults matching the specified values.

```
#include <clck.h>
```

Public Attributes

- int confidence = 0
- std::string id
- std::string node
- int severity = 0

4.13.1 Detailed Description

Suppress faults matching the specified values.

4.13.2 Member Data Documentation

4.13.2.1 int clck::Layer::Suppression::confidence = 0

Suppress all faults with a value less than the confidence value.

4.13.2.2 std::string clck::Layer::Suppression::id

Suppress all messages with a matching id. If empty, does not suppress on id.

4.13.2.3 std::string clck::Layer::Suppression::node

Supress all messages containing the node value. If empty, does not suppress on node.

4.13.2.4 int clck::Layer::Suppression::severity = 0

Suppress all faults with a value less than the severity value.

Index