Performance Considerations

Table of Contents

1. Introduction

Sakai generates events based on different actions performed by Sakai tools. These events are by default persisted to the SAKAI_EVENT table which grows very fast and may contains thousands or, most probably, millions of event entries. SiteStats aggregates these events into separate tables for faster tool access. This process, which implies event parsing and filtering, is expensive and some considerations must be taken into account. Specially for large deployments where there are many events generated per second, it is important to adjust SiteStats behavior for better performance.

2. Check for database table indexes

There is a bug in Hibernate (at least for MySQL) that prevents the correct creation of table indexes when creating the SiteStats tables with auto.ddl= true. Please check that all the indexes have been created and match the ones on these files:

- MySQL ddl
- Oracle ddl

3. Choosing the aggregator approach: realtime thread vs quartz job:

SiteStats procides two approaches for event aggregation:

3.1 Realtime thread (*default*)

This is the default option which is suitable for most small to medium deployments. However, it is up to the institution to determine if the number of events generated by sec are ok for using the realtime thread. At UFP, we have ~6000 active users, ~2000 logins per day and ~50000 events generated per day and we stick with the default realtime thread approach. This has the advantage of having all the statistic data up-to-date.
Relevant configuration properties:

- **`collectThreadEnabled`**`@org.sakaiproject.sitestats.api.StatsUpdateManager = true` (*default*)
- **`collectThreadUpdateInterval`**`@org.sakaiproject.sitestats.api.StatsUpdateManager = 4000` (*default*)

3.2 StatsAggregateJob quartz job

As an alternative, the StatsAggregateJob quartz job can be used. This has the advantage of scheduling the aggregation process to time periods where Sakai has lower load.
Relevant configuration properties:

- **`collectThreadEnabled`**`@org.sakaiproject.sitestats.api.StatsUpdateManager = false`
- **`maxEventsPerRun`**`@org.sakaiproject.sitestats.api.StatsAggregateJob = 50000` (*default*)
  Please take a look at all the related properties and their descriptions at this section

4. Event filtering

If not all sites have the SiteStats tool, events can be collected only for those sites using the property:

- **`collectEventsForSiteWithToolOnly`**`@org.sakaiproject.sitestats.api.StatsUpdateManager = true` (*default*)

Administrator generated events can also be discarded:

- **`collectAdminEvents`**`@org.sakaiproject.sitestats.api.StatsUpdateManager = false` (*default*)

Type of events to be aggregated can also be limited, if there is interest. The default configuration file is toolEventsDef.xml and can be overrided or, optionally, perform an union or subtraction of suported events using the properties outlined in this section.

5. Use an external database for SiteStats

This is extremely advisable for large institutions. Keeping SiteStats tables outside the Sakai database (even if on same server) heavily reduce Sakai DB impact.
Please look at this section for instructions on configuring an external database for SiteStats.
Note: this feature requires SiteStats version >= 1.2 or trunk

6. Using the latest version:

Please make sure to always have the lattest tag from SiteStats tool svn. Changes may include performance improvements.