Architecture book

by Mark Plummer

Table of Contents

This is documentation for project X.

Authentication

summary: "ThoughtSpot provides LDAP/AD, SAML, and ThoughtSpot login to authenticate users." sidebar: mydoc_sidebar permalink: /:collection/:path.html — ThoughtSpot provides three ways to authenticate users LDAP/AD, SAML, and ThoughtSpot login. In general, ThoughtSpot recommends that you use LDAP/AD or SAML if possible since ThoughtSpot provides only basic authentication with no restrictions on passwords, timeouts, failed logins, etc.

The following table shows each of the options and the items to consider for each.<table><tr><th>SAML</th> <th>LDAP/AD</th> <th>ThoughtSpot</th></tr> <tr><td><ul><li>Use SAML for single sign-on authentication.</li> <li>Can redirect from ThoughtSpot to SAML logins.</li> <li>Recommended for portal integration.</li> <li>Option to sync users and groups if stored in LDAP/AD.</li></ul></td> <td><ul><li>Configuration.</li> <li>Users authenticate against LDAP or AD.</li> <li>Option to sync users and groups with ThoughtSpot to manage group membership.</li></ul></td> <td><ul><li>User created and managed in ThoughtSpot.</li> <li>No enterprise password control (expiration, password strength, etc.).</li> <li>Only recommended when SAML and LDAP are not options.</li></ul></td></tr></table>

All users and groups must be known to ThoughtSpot. If you are using LDAP/AD or SAML and don’t create users in ThoughtSpot, a user is created when the user first logs in. However, this user is assigned to the `All` group and can only see content available for all users.

Groups are the primary way that security is managed. Groups are not automatically created. You can create groups and users manually or you must automate the assignment from a source system. ThoughtSpot has an assignment script that works with most LDAP / AD stores. It also has public APIs that you can use to sync users and groups between source systems and your ThoughtSpot appliance.

Architecture components

last_updated: tbd summary: "To implement ThoughtSpot it is important to understand where it sits within your overall analytics architecture and how it provides data to end users. " sidebar: mydoc_sidebar permalink: /:collection/:path.html — ThoughtSpot consists of a cluster of one or more nodes, acting together to provide analytic answers to business questions. As such, there are only a few integration points with ThoughtSpot on your network. The major components in a ThoughtSpot cluster are:

[architecture] | *{{ site.baseurl }}/images/architecture.png*

ThoughtSpot can handle a wide variety of different data sources. ThoughtSpot does all analysis against data in memory to help achieve fast results across millions and billions of records of data. ThoughtSpot caches the data in order to process it.

The ThoughtSpot appliance can be a physical appliance that ThoughtSpot ships, one or more AWS instances that are clustered together, or one or more VMware instances that are clustered together. From an external interface, regardless of the appliance type, the appliance appears to be a single instance.

For authentication (logging in), some source of user information is required. These define the login requirements and access control groups. Users must access the data from a supported browser to view saved content, or perform searched-based analytics. Finally, it is recommended that you have some sort of networked attached storage for storing backups in case of hardware failure.

Data Caching

last_updated: tbd summary: "ThoughtSpot does all analysis against data in memory to help achieve fast results across millions and billions of records of data." sidebar: mydoc_sidebar permalink: /:collection/:path.html — ThoughtSpot caches data as relational tables in memory. The tables can be sourced from different data sources and joined together. ThoughtSpot has several approaches for getting data into the cluster.

[DataCaching] | *{{ site.baseurl }}/images/DataCaching.png*

JDBC and ODBC Drivers

ThoughtSpot provides a JDBC and ODBC driver that can be used to write data to ThoughtSpot. This is useful for customers who already have an existing ETL process or tool, and want to extend it to populate the ThoughtSpot cache.

JDBC and ODBC drivers are appropriate under the following circumstances:

- have an ETL load, such as Informatica, SSIS, and so on
- have available resources to create and manage ETL
- have smaller daily loads

tsload

You can use the `tsload` command line tool to bulk load delimited data with very high throughput. Finally, individual users can upload smaller (< 50MB) spreadsheets or delimited files.

We recommend the tsload approach in the following cases:

- initial data load
- JDBC or ODBC drivers are not available
- there are large recurring daily loads
- for higher throughput; this can add I/O costs

Choosing a Data Caching Strategy

The approach you choose depends on your environment and data needs. There are, of course, tradeoffs between different data caching options.

Many implementations use a variety of approaches. For example, a solution with a large amount of initial data and smaller daily increments might use tsload to load the initial data, and then use the JDBC driver with an ETL tool for incremental loads.

Performance considerations

ThoughtSpot configuration and licensing varies by memory availability. Other considerations also impact the performance of your solution. Because some solutions perform better than others, think about the following issues before implementation.

Nodes that have 250GB of memory capacity perform optimally with less than 250GB of data, and less than 250 million rows of data in each node in a ThoughtSpot cluster. Smaller nodes, like nodes with 200GB memory capacity, serve proportionally smaller loads.

To optimize performance for more complex schemas, we recommend fewer rows of data for each node. To reduce the total amount of data and rows of data, we recommend the following approaches:

- Limit the data range to the relevant years or months.
- Combine long and narrow tables into wider tables whenever possible.

Data Boundaries

Total rows in a result of a join can have an impact on performance. In general, we recommend that you have fewer than 10 billion rows in a many-to-many join. Also, consider these boundaries:

Contact ThoughtSpot support for guidance on boundaries for the following:

- Maximum number of rows that can be downloaded
- Size in CSV format
- Total number of rows across all tables
- Many-to-Many (Generic join cardinality)
- Load frequency

Worksheet Boundaries

Worksheets must have less than 1000 columns. For aggregated worksheets, follow these guidelines:

- Number of columns should be less than 50
- Number of rows should be less than 10 millions

You can use an ETL process to circumvent these limitations. Speak with ThoughtSpot support to learn more.

Aggregated Worksheets and Joins

To join an aggregated worksheet with a base table, you must configure your installation to allow this behavior.

- The aggregated worksheet cannot use more than 5 component tables.
- The number of rows in the final aggregated worksheet cannot be greater than 1000.

Chasm Trap Worksheets

For chasm trap scenarios where two or more fact tables join through a shared dimension, we recommend the following boundaries:

Description	Boundary
Maximum number of fact tables in a worksheet	5
Maximum number of shared dimensions	2
Maximum number of rows in a *non* co-sharded shared dimension table of chasm trap	1B
Maximum number of rows in a co-sharded shared dimension table of chasm trap	1B

Row-level Security Boundaries

Maximum number of unique RLS rules with search data suggestions should not exceed 15K.

Scheduled Pinboards

For ideal performance of scheduled pinboards, do not exceed 50 scheduled pinboard jobs.

Data and object security

Object Security

Object security controls what content users see within ThoughtSpot. Objects are tables, columns in tables, worksheets, pinboards, and saved answers.

Users gain access to objects when an object owner shares access with them. Owners can share with individual users or with entire groups, giving access to everyone within that group. Objects may be shared with edit or view-only options. A user can automatically share objects with anyone else in the groups to which they belong. This has implications on setting up privileges, and on applying row-level security.

Permissive Security Mode

The default Permissive Security mode of ThoughtSpot means that when someone shares an object with you, you can see all the data it uses, regardless of explicit permissions to the parent object data. You can see a shared pinboard without having access to its underlying worksheet or table.

Advanced Security Mode

ThoughtSpot's Advanced Security mode is opposite of the default permissive mode. Unless the user has explicit permissions to the entire stack of parent objects, they cannot see the data in the child object. For example, in a shared pinboard, you can see data only if you have explicit permissions to the relevant columns of the parent worksheet. Similarly, you can only see the data in a worksheet to which you have access if you have explicit permissions to its parent table object.

Work with your ThoughtSpot support team to enable the Advanced Security Mode on the relevant clusters.

Row level security (RLS)

Row level security controls what data a user can see in each shared piece of content. Even if a user has access to a worksheet, they can only see rows from the tables they have permission to see.

RLS applies at the table level, so it automatically extends to all worksheets, saved answers, and pinboards based on that table, every time. Also, in queries where there are tables with table filters, all joins are always enforced to avoid accidentally allowing users access to data they shouldn't see.

RLS requires three things:

- A table filter with a column (possibly in a joined table) that can be used to determine who can see a row, such as account id or tenant id.
- A group that can be associated with the row of data by name. For example, if the column is `account_id` and has values of `1`, `2`, `3`, users can be assigned to groups `group_1`, `group_2`, `group_3` and then only see their data.
- Users must be assigned to the group. If they are not assigned to a group that has access, they do not see any data.

Administrative users can always see all rows of data because RLS does not apply to them.

RLS supports a hierarchy of groups, which makes it possible to grant access to some users across multiple groups.

Keep in mind that users within a group can share with one another. If you put everyone in your organization into the same group for RLS, they can share with anyone in the company.

Column level security (CLS)

Column level security lets users see certain columns in a table, but not other columns. This can be accomplished by sharing a limited set of columns in a table with specific users or groups.

Because someone can share with anyone in the same group, they can potentially share restricted columns. For example, if a *Human Resources* repository has a column with salary information, and it appears in a worksheet, any *Human Resources* group member could create an answer with visible salary information and mistakenly share with someone outside of *Human Resources*. That 'outside' person now has access to the salary information. In such cases, we recommend that you work with your ThoughtSpot support team to enable the Advanced Security Mode on the relevant clusters.

System privileges

System privileges refer to what a user can do in ThoughtSpot. For example, can they upload or download data or share with all users. These privileges are defined on a group level and inherit downwards. So, if Group A had child groups Group B and Group C, then any privilege given to Group A is also available to Group B and Group C. What this often means is that separate sets of groups are required to manage privileges.