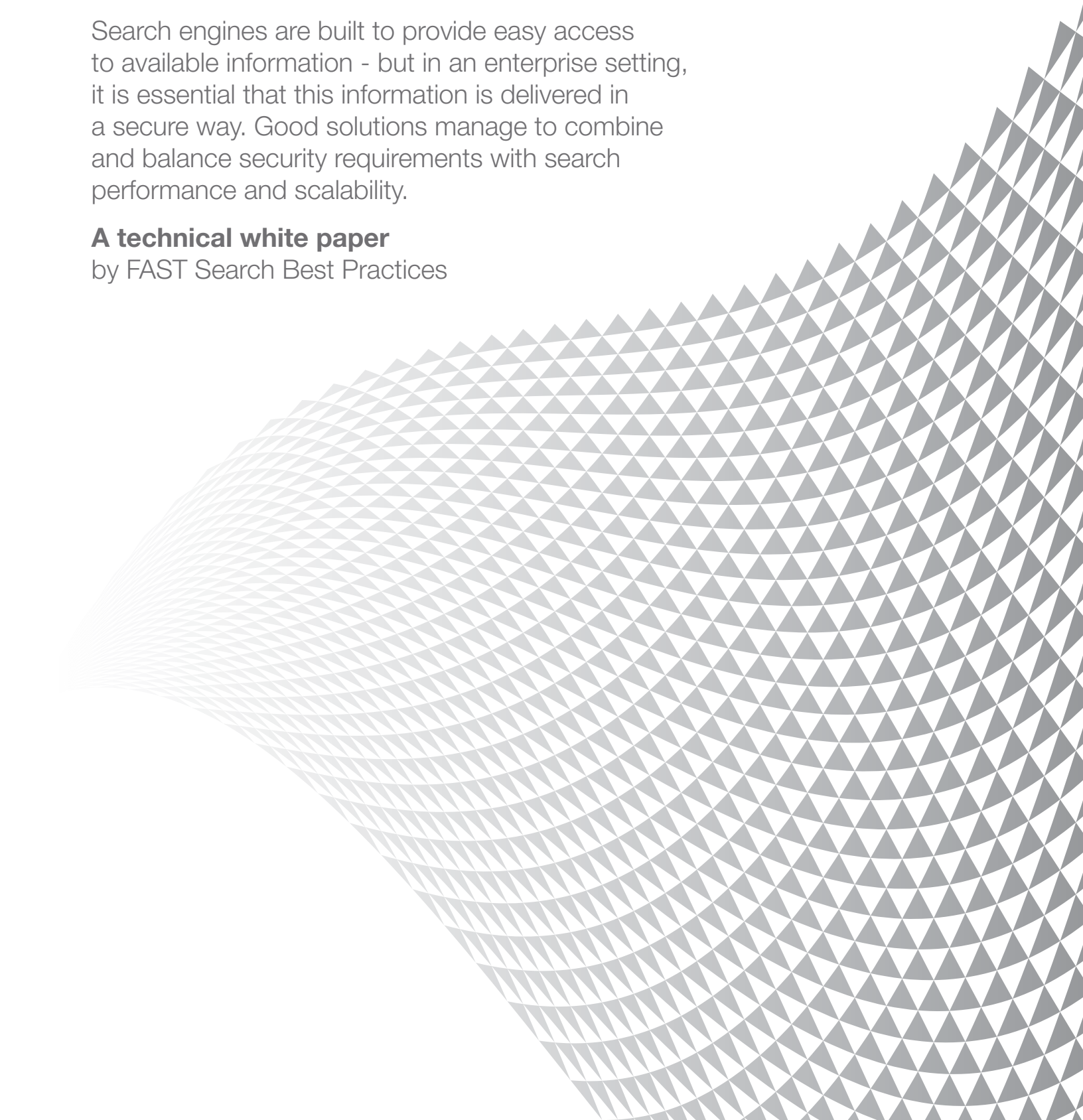
fast SBP™

Secure search

Search engines are built to provide easy access to available information - but in an enterprise setting, it is essential that this information is delivered in a secure way. Good solutions manage to combine and balance security requirements with search performance and scalability.

A technical white paper

by FAST Search Best Practices



5 things you should know about security

1. A system is only as secure as its weakest link. Don't spend time and money on traffic encryption and yet allow staff to leave print-outs of confidential data lying around
2. Document-level security means that individual documents cannot be accessed by other authorized users of the system
3. System-wide security means locking down against unauthorized access using encryption, IP filtering, and OS-level security
4. Index-based ACL mapping is faster and more scalable than post-query filtering
5. Search engines typically rely on being installed within a secure environment to maintain complete data integrity

The goal of a search engine is to provide easier and better access to information, whenever and wherever that information may be important. Yet much of that data may be confidential. Although a search engine is a gateway to sensitive corporate data, it also acts as a gatekeeper for such data. So the search engine must be thought of as a trusted computing base.

Security is applied to three areas. First, in terms of managing end users, it is used to verify their identities and the levels of content that they're entitled to access. Second, from an application perspective, security validates that all Application Programming Interface (API) calls are issued by authorized clients and that connectors are respecting each repository's correct access model. Third, security is used to manage the authorization control of the administrators who modify the search system itself.

A search solution must be integrated into the security fabric of the organization or of the host IT shop

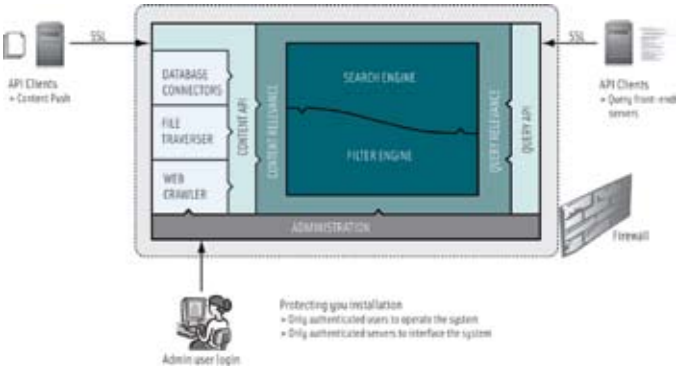
All of these elements have links to search; the most critical is that the permission levels for documents when executing a query are enforced by search software. For example, within an OEM environment, it's crucial to ensure that access via the search interface provides the same level of security as the actual application.

The same challenge occurs when connecting to a third-party application such as a Document Management System (DMS) where the access control model must be reverse-engineered.

The context for document-level security

It is essential to the maintenance and scalability of an organization's IT systems that different employees all use the same IT infrastructure and servers. This may seem obvious, but good security relies on all the systems in use having the ability to segregate and protect data, and giving document access only to those who are entitled to it.

Some Elements of Search Security



This is particularly sensitive with respect to "Chinese Walls" (barriers between employees to avoid conflicts of interest) and to other confidential data such as employee salaries. Using folder and document-level access control within applications is the most common corporate practice to ensure appropriate privacy. This access control logic must then be respected by other applications connecting to the content, including search engines.

One solution is to use index-based access filtering, where the search engine indexes Access Control List (ACL) information along with each document, and resolves permissions at query time, acting as a sophisticated metadata filter. Users only see documents within a hit list that they have sufficient credentials to view. This is the most scalable solution, but only some search engines provide this functionality.

Q: I have Exchange and Documentum with NT usernames and Lotus Notes which use their own conventions. Can I still do secure search in one pass across all three sources?

A: Assuming an index-based security solution, each document contains an ACL which will contain users and groups in the repositories' own formats. The tricky part is for the front-end application to determine the Domino and NT names for a user once he or she has been authenticated. This information will typically be stored within an LDAP-compliant directory or a SSO product. Once that information has been obtained, the front end can find out what groups the user belongs to in each domain and then send all that information to the end user, together with the query text.

To begin with, the connector must be able to extract ACL information from the document repository. This information will describe the users and groups that have different access levels – who has read and write access, who has read-only, and who is denied access. In the case of an NT file system, this access information is contained within the operating system metadata, whereas within a DMS, an API call normally allows the data to be extracted for each document. The ACL is then processed and normalized before being added as special metadata within the search index.

The query-side application determines which groups a user belongs to, and group resolution will be performed across multiple repositories if necessary. The search solution must either maintain a cache or have access to an up-to-date mapping of this information. The other solution is to resolve the user-to-group mapping at index time. This is not recommended as the ACLs will potentially grow very large, and each time a group is modified the ACL must be re-indexed. This will cause a large numbers of document updates.

Q: I am integrating search into my secure application. The push mechanism for indexing is using 128-bit encryption to secure the data. Is that safe enough?

A: N-bit (such as 128-bit) encryption means that the key required to decipher the encoded data has a length of N. For brute-forces attacks, N is a measure of how difficult an encryption algorithm is to crack, since it determines how many permutations must be tried in order to find the correct key. For example, even if a computer could test one trillion keys a second, it would take two million million million years to decipher a 128-bit key. It is assumed that for at least the next 10 years, 128-bit encryption is virtually unbreakable.

After the above steps (extracting ACLs and resolving group membership) in an index-based strategy, the search engine is responsible for resolving and enforcing the actual access control.

Instead of embedding the access permission information in the search index, an alternative way to handle document-level security is to check entitlement in real-time against the source. For each hit, the search application will check the user's access rights against the repository.

By and large, this solution does not scale since the front end will have to retrieve a potentially huge number of results for each query to find enough authorized documents. This also adds load to the document repository. An intermediate solution is to combine the two methods. The search core can filter the results, perform a final real-time check to verify that a user's permission has not been downgraded, and filter out documents for which the user has lost privileges. This will increase security in between index updates at the expense of query performance. (If the feeding process and index freshness are correctly configured, this should not be required.)

Location sensitivity is built into some systems. This is modifying security at point of login where the access model for content is modified or overridden based on where and how a user connects to the search environment. For example changing access depending on connection type (public, DMZ, behind firewall, wireless, wired) and device type (workstation, laptop, PDA, cell phone). These complex rules must be reflected in the search engine, and may require a hybrid approach.

Collection-level security may also be used. Here, the application tier will assign different authorization levels to various collections within the search index. End users will then have access to the set of collections that map to their authorization levels.

Searching within a secure environment

Document-level security is the most important element of deploying a search engine in a secure installation. Other areas to consider when searching within a secure environment are protecting data access and transfer to and from the search engine.

Assuming that the search engine is returning protected information, the connectors must have access to all secure data and must authenticate to the applications sourcing it as highly privileged users. A wide range of security protocols are in use among all potential sources, although using connectors hides many of the application-specific details such as native APIs. The other option to get content into a search index is for the repository to push data, rather than having connectors pull data. In principle, this does not change the requirements for authentication between the application and connector, although the specific details change.

Server-level security is required in order to protect the integrity of the trusted computing base itself; it can be accomplished using firewalls, and by ensuring that all traffic, within and through the firewall (query, content feed, and administrative access) uses appropriate encryption and security protocols.

In addition, user authentication must be enforced to ensure that only authorized individuals are granted access. Typically, the search interface is embedded within an application or portal that performs this authentication. One set of authorizations govern search user access to collections and documents. Another set of authorizations governs administrative user access to various administrative functions, although the issues of user authentication and authorization are much the same. It's recommended that even if query access is not restricted (such as in an e-commerce or Web search scenario), authentication always applies to the administration console.

Designing a secure system

Security is a complex topic. When designing a secure system, the first rule is to plan ahead, to understand what the users really need and what the corporate IT infrastructure can realistically accomplish. For example, a single sign-on (SSO) solution is generally recommended for knowledge management projects, or where the search index spans multiple authorization-controlled repositories. However, if the merging and unifying of authentication and usernames across all repositories is not already in place, the search roll-out may be delayed if it is waiting for an SSO implementation.

Integrating the search system with the corporate central security directory (i.e. Microsoft ADS, LDAP or Netegrity) is also recommended for seamless and secure document access.

Collection-level security can be used when there is a division of data without much granularity. It is useful for departmental separation of information in a company, or when the underlying repository has no security. For instance, there may be two servers for shared documents, one for marketing and the other for finance, where the separation is enforced by common usage rather than Active Directory. Collection-based security could be chosen to enforce this practice.

Choosing the correct document security model

Once the security model of the application has been determined, the focus becomes the correct replication of all underlying access control mechanisms.

When reverse-engineering of access control models is not possible or is undesirable because of the system's complexity, or an index-based solution is unacceptable due to the security latency, the best recommendation is to perform post-filtering of the search results against the source to remove unauthorized documents. For instance, dynamic access managers (using the time of day, the client's IP address, or the strength of the authentication mechanism to govern access to data) require degrees of sophistication that may be prohibitive to ACL mapping.

However, with this approach, when a user requests N hits, the filter will often need to request much more to account for those that will be removed. In particular, as the number of documents in the index grows and the ratio of documents that each person can see diminishes, query performance will drop. Additionally, even for small indices, performance is often worse because of the time needed to submit requests to the source. Therefore, the index-based approach to document security is more scalable.

If there is no way a connector can extract access control information on a per-document basis (for example, if it is not supported by the underlying API) it will be necessary to fall back to a query-time filtering approach. Caching is then used to increase the speed of searching. Generally, this is worth the extra engineering effort because a user will often search for related themes where the same documents appear in hit lists or repeat the search to review the results. The benefits of caching user-to-document matches will therefore be apparent.

If the preferred approach of index-based authorization is used, there should be no issues for the integrator or search development team in the case of a connector purchased with built-in document-level security. On the other hand, for custom connectors and push mechanisms, where the application developer is responsible for designing the security, it's important to scrutinize the ACL model. There will inevitably be caveats and individualities from one source to another which must be studied for both the query logic and the ACL creation at document processing time.

Results granularity need also be considered, where the document is not the atomic unit of security. A proper security model should support property-level security, i.e. which fields can be viewed.

The final element to the index-based method is building the user-to-group mapping. This can be cached within the application session to avoid slowing down queries. If a further update latency is acceptable in exchange for faster login times, an external cache can be updated on a schedule.

The downside of index-based mapping is the lag between an ACL being updated and the search index being notified. This is usually acceptable, as the lag period will actually be very short. In addition, the false positives returned will be documents that the searcher was able to view during the last update cycle.

However, some system managers feel that this represents a security hole. It may be unacceptable for users whose document-access privileges have been revoked to see a document's title and teaser within a hit list. In such circumstances, a mixed approach is recommended, with a last-minute check performed in real-time. This approach benefits from some of the scaling advantages of the mapped ACL solution along with the real-time validation of a post-query filter.

Mini case study
Entertainment portal rolls out scalable secure search

Who
Italy's largest information and entertainment portal.

Challenge
To allow secure search across a wealth of data from many sources, including 4 million intranet documents and 6 million crawled documents with over 20,000 user sessions per hour.

Solution
Multiple connectors (Lotus Notes, File Traverser) all indexing ACL information from the source, with connector surveillance to monitor for ACL updates that can be quickly pushed to the search index. In the front end, a Single Sign-On is used for document retrieval. The application also links to both Active Directory and Lotus Notes for user-to-group resolution.

The fundamental steps for improving search

There are two key takeaways regarding search and security. First, index-based ACL resolution is faster and more scalable than post-query results filtering. Second, search engines are only as secure as the firewall and the user authorization infrastructure they reside behind.

Developers sometimes attempt to break a search engine's security mechanisms, since the time taken to crack a system is seen as a measure of its integrity. In fact, an IT manager's main concern will be the security surrounding the search engine. For the engine to function properly, it requires only a very limited amount of access from third-party applications. Therefore, the way to secure a search engine is to deny access to it, which entails:

- Putting all search engine components on the same network behind the same security infrastructure
- Securing the hosted systems by IP address and port
- Encrypting communication between the calling application and the query server.

In this way, the search engine simply becomes another server that needs to be protected, but will not open any security holes in the company's IT infrastructure.

Search security must work in conjunction with other IT security mechanisms and policies.

When designing secure search, the principal goal is that correct document level security must be ensured, but it must be imperceptible from a performance and scalability perspective. It needs to meet these criteria:

- The search provider must feel confident that no information will leak
- Users need to be assured that no unauthorized people will see their data.
- IT managers must be able to support search for multiple user groups within the same IT infrastructure
- Business managers need assurance that the search technology used will enforce the corporate security policies.

With a well-designed and well-planned index-based security model, all of these criteria are attainable without compromise to search speed or scalability. That way, every stakeholder can feel confident about the integrity of the search application.

Frequently asked questions

Q: What is AD?

A: Active Directory (AD) is a Microsoft service that identifies all resources on a network and makes them accessible to users and applications.

Q: What is LDAP?

A: The Lightweight Directory Access Protocol (LDAP) is a set of protocols for accessing information directories. AD is an example of an LDAP-compliant directory.

Q: What are an ACL and a DACL?

A: An Access Control List (ACL) is a set of data that tells an operating system or application what access rights each user has for an object. A DACL (Discretionary ACL) is a user-controlled ACL.

Q: What does encryption do?

A: An encryption algorithm modifies data so that it's unreadable to applications except those for which it is intended. Decoding the information requires knowledge of the algorithm and either one or two (public and private) keys.

Q: Can an index-based ACL solution show users documents that they don't have permission to see?

A: No. In certain circumstances users may see the title and teaser of a document they previously had permission to see (their permission may since have been removed.) They will not be able to see the actual document, though, since the underlying application will perform its own authorization check when requesting the original.

Q: What are Chinese Walls?

A: Chinese Walls are the internal policies put in place, typically in financial organizations, to restrict communication between different teams within the same company. The goal is to segregate knowledge transfer where a conflict of interest is possible, for example, between teams working on different sides of a same deal.

About FAST SBP™ (Search Best Practices)

SBP consulting is a highly focused transfer of search knowledge and experience from FAST to its prospects and customers. SBP workshops aim to help enterprises realize the full potential of search, by creating optimal strategic, functional and technical roadmaps, delivered in the form of business model, solution and architecture designs.

For any feedback or questions related to this paper, please contact us at sbp@fastsearch.com.

Fast Search & Transfer
www.fastsearch.com
info@fastsearch.com

Regional Headquarters

The Americas
+1 781 304 2400

Europe, Middle East & Africa (EMEA)
+47 23 01 12 00

Japan
+81 3 5511 4343

Asia Pacific
+612 9929 7725

SWP.010.T.01.020806