Google Search Appliance

Managing Search for Controlled-Access Content

Google Search Appliance software version 7.4

Google

Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
www.google.com

GSA-SEC_200.03
March 2015

Contents

Chapter 1

Overview

The Google Search Appliance makes documents in your domain discoverable through search. In addition to public content that is available to everyone, the search appliance can crawl and index documents that require a login and password or another form of authentication. To protect confidentiality at serving time, the search appliance determines whether the user performing the search is authorized to view each document before it displays results.

For instance:

- You have several sign-on domains, and want to enable employees to search across enterprise content without logging in to many sites.
- You want to make an article searchable by everyone, but to require that a user supplies a login before they can view the full text.
- You want to allow the finance team to search through confidential reports on an accounting site. Members of the finance group can search for reports and read them. Employees in other divisions cannot view accounting reports and should never see these documents in their search results.

As the search appliance administrator, you must configure the Google Search Appliance to support these kinds of situations.

About this Guide

This guide is intended for the search appliance administrator and developers who need to understand authentication and authorization for the Google Search Appliance. It explains how the Google Search Appliance makes controlled-access content available through search, describes how to configure authentication and authorization, and demonstrates how to make controlled-access content available to authorized users in your organization.

Which Sections of this Guide Should I Read?

This guide helps you to answer the following questions:

- How do I set up my search appliance to crawl and index controlled-access content?
- Once I have indexed controlled-access content, how do I specify the content that is visible to a user during serve? Public content (`access=p`) is available in all search results, while secure content (`access=s`) is only visible to authorized users.

Because some methods of accessing controlled-access content do not support secure serve, the answers to these questions depend on your existing access control infrastructure, and whether your content sources require secure serve.

The following table explains which sections in this guide are most relevant for each access method, and provides links to those sections.

Access Method	Access Type	Suggested Crawl Method	Suggested Serve Method
HTTP Basic or NTLM HTTP	Public or secure	Crawler Access (see "Configuring Crawl for HTTP Basic or NTLM HTTP" on page 12)	HTTP Basic or NTLM authentication (see "HTTP-Based Authentication" on page 25)
Access content on a Windows or SMB/ CIFS file share	Public or secure	Crawler Access (see "Configuring Crawl for HTTP Basic or NTLM HTTP" on page 12)	Pass user credentials and optionally authenticate with LDAP (see "Integrating the Search Appliance with an LDAP Server" on page 36)
Single login domain: Windows (Kerberos) Authentication for Windows Server or Sharepoint Server	Public or secure	Crawler Access (see "Configuring Crawl for HTTP Basic or NTLM HTTP" on page 12)	IWA (Integrated Windows Authentication) / Kerberos authentication (see "Kerberos-Based Authentication" on page 28)
Single login domain: one set of domain credentials provides access to all content, and the login form does not use frames or JavaScript.	Public or secure	Forms Authentication (see "Configuring Crawl for Cookie-Based Access" on page 11)	Cookie-based authentication (see "Cookie-Based Authentication" on page 23)
Single login domain: one set of domain credentials provides access to all content. The login form is plain HTML. Single or multiple cookie domains.	Public or secure	Forms Authentication (see "Configuring Crawl for Cookie-Based Access" on page 11)	Cookie-based authentication (see "Cookie-Based Authentication" on page 23)

Access Method	Access Type	Suggested Crawl Method	Suggested Serve Method
Multiple login domains: more than one set of credentials are required to provide access to all content.	Public or secure	Forms Authentication (see "Configuring Crawl for Cookie-Based Access" on page 11)	Cookie-based authentication (see "Cookie-Based Authentication" on page 23)
Multiple login domains: more than one set of credentials are required to provide access to all content.	Public or secure	Crawler Access (see "Configuring Crawl for HTTP Basic or NTLM HTTP" on page 12) or Forms Authentication (see "Configuring Crawl for Cookie-Based Access" on page 11)	Mixed authentication mechanisms (see "Configuring Credential Groups" on page 22)

For information about specific secure search limitations, see Specifications and Usage Limits.

Chapter 2

Crawl, Index, and Serve

This chapter describes how a search appliance discovers content on your servers. It provides an overview of authentication and authorization methods used during crawl and index, and the methods available during serve. It also provides basic instructions for configuring a search appliance to crawl, index, and serve controlled-access content.

Authentication, Authorization, and Controlled-Access Content

Authentication is the process of verifying the identity of a user, a system, or a service. Authorization is the process that determines whether an authenticated user, system, or service has permission to perform a task. The term "controlled-access content" represents any information that should not be displayed unless the user who requests the content is authenticated and has authorization to view the information.

To make controlled-access content discoverable through search, the search appliance mediates two kinds of access:

- Access that enables the crawler to discover content on your servers and index any controlled-access content found there.
- Access that enables an individual user to perform a search and to view content that exists in the index.

All controlled-access content that is available to the search appliance is indexed. For more details, see "Crawl and Index for Controlled-Access Content" on page 10. After the controlled-access content is indexed, the search appliance determines whether to display the content in response to each search request.

When a user issues a search request for content controlled by some authentication mechanisms, the search appliance impersonates the user. The search appliance verifies the user's identity and determines whether the user has authorization to view controlled-access content. This check is performed before the search appliance displays any content in search results.

The Google Search Appliance provides centralized serve-time authentication with Universal Login (see "Universal Login" on page 17). With centralized serve-time authentication, a user who is searching for protected content is prompted for credentials once by the **Universal Login Form** for set of authentication mechanisms that share a username and password. For detailed information about Universal Login and authentication, see "Authentication" on page 17.

After the search appliance authenticates a user by establishing the user's identity, the search appliance performs authorization checks to determine whether a user has access to the secure content that matches their search. For detailed information about authorization on the Google Search Appliance, see "Authorization" on page 41.

A Google Search Appliance provides additional methods for enabling authentication and authorization that do not require user impersonation. These are discussed in "The SAML Authentication Service Provider Interface (SPI)" on page 34 and "How to Exclude Controlled-Access Content Sources from Search" on page 48.

This chapter provides information on how to configure the Google Search Appliance to crawl, index, and serve controlled-access content. For examples of configuring a search appliance, see "Use Cases with Public and Secure Serve for Multiple Authentication Mechanisms" on page 52.

Crawl and Index for Controlled-Access Content

The Google Search Appliance indexes all content that can be crawled and indexed. This includes both controlled-access content and content that is available to anyone. Once you set up the search appliance with access credentials, it will maintain a copy of all crawled content in the index. The index allows the search appliance to determine relevance and display secure results when a user performs a search. Users only see the secure results that they are authorized to view.

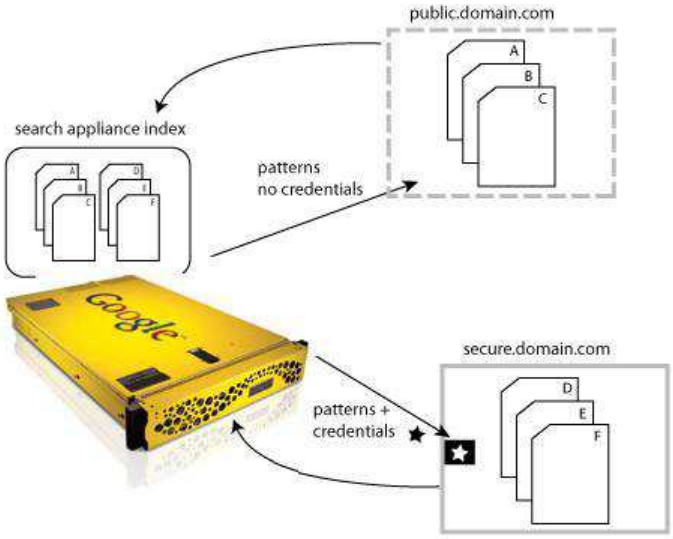
How a Search Appliance Indexes Controlled-Access Content

A search appliance discovers and indexes controlled-access content in the same way that it indexes all other content: by performing a crawl through the content sources that are available to the web crawler, file system crawler, relational database crawler, and the XML content feed interface.

When you define content sources, you must perform additional steps in the Admin Console to give the search appliance access to controlled-access content:

1. Provide the search appliance with URL patterns that match the controlled content.
2. Give the search appliance access credentials to use with those patterns.

You can specify a different set of access credentials for each URL pattern in the Admin Console. The means by which you provide these credentials is different for each kind of authentication, but the general process remains the same.



Configuring Crawl for Cookie-Based Access

The search appliance supports cookie-based access (single sign-on, forms). For sites that require the use of a cookie for authentication during crawl and index, you can define your content with a forms authentication rule. When you set up the search appliance to crawl cookie-based content, consider the following points:

- Define a rule under **Content Sources > Web Crawl > Secure Crawl > Forms Authentication** for controlled-access content sources that require the search appliance to obtain a session cookie from a login form. Content accessed through a forms authentication site can be secure or public during serve. For more information click **Admin Console Help > Content Sources > Web Crawl > Secure Crawl > Forms Authentication** in the Admin Console.
- If the URL pattern that matches the forms authentication rule includes a logout page, the search appliance attempts to crawl the logout page, which essentially results in cookie expiration. If the SSO system includes a logout page, then exclude the logout page by adding it to **Do Not Follow Patterns** on the **Content Sources > Web Crawl > Start and Block URLs** page. For more information click **Admin Console Help > Content Sources > Web Crawl > Start and Block URLs** in the Admin Console.
- A forms authentication rule must generate at least one action for the search appliance to consider it valid. If a rule doesn't generate any action for a URL, the search appliance logs an error and doesn't crawl the URL again.

Google has certified the following Single Sign-On systems for use with software release 6.2 and later:

- Computer Associates SiteMinder 6.0, Policy Server and Web Agent
- Oracle Access Manager 7.0.4 (formerly Oblix)
- Cams by Cafesoft, version 3.0

Configuring Crawl for HTTP Basic or NTLM HTTP

When you set up the search appliance to crawl controlled-access content with HTTP Basic or NTLM HTTP, consider the following points:

- The Crawl and Index process for content that uses HTTP Basic and NTLM HTTP is controlled by parameters under **Content Sources > Web Crawl > Secure Crawl > Crawler Access**. To learn more about setting up crawl for HTTP Basic and NTLM HTTP, click **Admin Console Help > Content Sources > Web Crawl > Secure Crawl > Crawler Access** in the Admin Console.
- If you are using HTTP, Google recommends that you use HTTPS for all requests for controlled-access content because HTTP Basic passes user credentials as clear text. To force the search appliance to perform crawl, index, and serve over HTTPS, see "Protecting the User's Credentials for Serve with HTTP Basic and NTLM HTTP" on page 39.

Configuring Crawl for HTTP Basic/NTLM and Cookies

The search appliance supports crawling content sources protected by both HTTP Basic/NTLM and cookies. To enable crawl of a site that's protected by this mechanism, you need to configure both forms authentication crawling and crawler access.

To configure crawling for a site that is protected by Basic/NTLM and cookies:

1. On the **Content Sources > Web Crawl > Start and Block URLs** page, add the URL for the protected site to the **Start URLs** and **Follow Patterns.**
2. On the **Content Sources > Web Crawl > Secure Crawl > Forms Authentication** page, create a new forms authentication rule for the protected site:
   a. For the **Sample Forms Authentication protected URL**, enter the URL of a content page in the protected site.
   b. For the U**RL pattern for this rule**, set it to the URL that you entered in the **Follow Patterns** or a more general pattern.
   c. Click **Create**.

      Normally when setting up Forms Auth, the Admin Console will render a login page for you to enter crawl-time credentials. But because there's no form associated with the URL, the Admin Console displays an error page generated by the content provider instead.
   d. Ignore the error page and click **Save and Close**.

      The Forms Authentication page appears and your new rule is listed with its pattern, action, and form fields.
   e. Click **Save**.
3. On the **Content Sources > Web Crawl > Secure Crawl > Crawler Access** page, set up crawler access:
   a. Create a new rule for the URL that you entered in the **Follow Patterns**.
   b. Click **Save**.

Configuring Crawl for the SAML Authentication and Authorization Service Provider Interface

Before using the Authentication and Authorization SPI, you must configure the appliance to crawl and index some secure controlled-access content. The SPIs are only used when a user queries for secure results. For content protected by the Authentication and Authorization Service Provider Interface, you can crawl secure content through HTTP Basic, NTLM HTTP, or with Forms Authentication:

- Make sure that you have defined some patterns for crawling your controlled-access content under **Content Sources > Web Crawl > Start and Block URLs**.
- For content that requires HTTP Basic Authentication or NTLM HTTP credentials, set up the crawl under **Content Sources > Web Crawl > Secure Crawl > Crawler Access** and clear the **Make Public** checkbox for at least one URL pattern.
- For content that requires a Forms Authentication rule to authenticate using a single sign-on (SSO) server, set up the crawl under **Content Sources > Web Crawl > Secure Crawl > Forms Authentication** and clear the **Make Public** checkbox for at least one URL pattern.

Configuring Crawl and Serve for Kerberos

The search appliance supports Integrated Windows Authentication/Kerberos authentication for both crawling and serving controlled-access content. Before you can configure Kerberos crawling, the search appliance must be configured to use Kerberos authentication at serve time. For information about configuring Kerberos-based authentication for serve, see “Kerberos-Based Authentication” on page 28.

After Kerberos-based authentication for serve is configured, you can enable Kerberos crawling by using the **Content Sources > Web Crawl > Secure Crawl > Crawler Access** page. For more information about enabling Kerberos crawling, click **Admin Console Help > Content Sources > Web Crawl > Secure Crawl > Crawler Access** in the Admin Console.

Providing Dynamic Serve-Time Security on a Per-URL Basis

The search appliance can accept a serve-time security setting for a document sent by a web server through the `X-Gsa-Serve-Security` HTTP header. This HTTP header can be useful for setting serve-time security for documents fed in by metadata-and-url feeds, as with Connectors V4.

The `X-Gsa-Serve-Security` HTTP header can only have one of the following values:

- `secure`--To mark a document as protected, use `X-Gsa-Serve-Security: secure`
- `public`--To mark the document as public, use `X-Gsa-Serve-Security: public`

The `X-Gsa-Serve-Security` HTTP header works along with other access-control mechanisms, including ACLs and x.509 certificates. For example, to ensure that a document is secure at serve time, you might provide:

- x.509 certificate security at crawl time, but no `X-Gsa-Serve-Security` HTTP header
- x.509 certificate security at crawl time and an `X-Gsa-Serve-Security` HTTP header with a value of `secure`
- A per-URL ACL and an `X-Gsa-Serve-Security` HTTP header with a value of `secure`

To ensure that a document is public at serve time, you might provide an HTTP header with a value of `public` with no ACLs or crawl-time security.

Configuring Crawl and Serve Over HTTPS

The search appliance uses digital certificates when communicating with web browsers and servers over HTTPS. The search appliance also supports the use of digital certificates to perform X.509 certificate authentication to verify a user's identity before serving secure results, as described in "Client Certificate-Based Authentication" on page 27.

To use HTTPS for all requests for controlled-access content, configure the search appliance to enable certificate use. The digital certificate for the search appliance must be recognized by other servers, and the certificate authorities for all HTTPS-secured sites must be valid (that is, must not be out of date and must be for the designated server name).

By default, the search appliance uses its own store of preloaded certificate authorities. These default certificate authorities are used by most browsers. By using these default certificate authorities, the search appliance trusts the same servers that browsers trust. As a search appliance administrator, you have the following options:

- Using the default certificate authorities without uploading any of your own certificate authorities
- Using only your uploaded certificate authorities without using the default ones
- Using both the default and uploaded certificate authorities

By using the options in the **Default Certificate Authorities** area of the **Administration** > **Certificate Authorities** page, you can disable or re-enable default certificate authorities. For information about using certificate authorities, click **Admin Console Help > Administration** > **Certificate Authorities**.

This section provides a general overview of how to install a digital certificate for use by the search appliance. For more detailed instructions, including an explanation of how to request a digital certificate from a certification authority and decrypt an encrypted private key, click **Admin Console Help** > **Administration** > **SSL Settings** in the Admin Console.

Note: The SSL Settings page can only install non-encrypted RSA keys in `.pem` (privacy enhanced mail) format. If the private key is encrypted or in PKCS#12 format (see http://en.wikipedia.org/wiki/PKCS12), refer to the instructions in the Admin Console Help.

To configure the search appliance to enable crawl and serve over HTTPS:

1. Log in to the Admin Console.
2. Choose **Administration** > **SSL Settings**.
3. On the **SSL Settings** page, scroll down to **Install an SSL Certificate**.
   - Under **SSL Certificate**, enter the file name of the certificate or click the Browse button to locate it. If you are using an intermediate certificate, enter the name of the file that includes both the intermediate certificate and the host certificate.
   - Under **SSL Private Key**, enter the file name of the unencrypted private key or click the Browse button to locate it. If the SSL Certificate contains an intermediate certificate, use the private key that corresponds to the host certificate.
4. Click the **View Certificate Information** button.
5. Installing the certificate will restart the Admin Console and the front end. If you are ready to install, click the **Install SSL Certificate** button.

   When the page refreshes, the following message appears at the top:

   ```
   SSL certificate installed. The appliance console needs to be restarted, please
   log in again.
   ```

6. On the **Admin Console** login page, click **Log in**, and log in using the admin username and password.
7. Choose **Administration > SSL Settings**. Your new certificate information is listed under **Current SSL Certificate Information**.

Secure Content and Public Content

Once controlled-access content is present in the index, the search appliance labels it as "secure" or "public":

- If the content is labeled "public," any user with access to the search appliance can view links to content in response to a search query.
- If the content is labeled "secure," the search appliance must authenticate the user and verify that the user has authorization to view the content before the search appliance includes links to the content in the search results.

It's important to understand that when controlled-access content is labeled as "public" in the index, it is shown in all users' search results. Because public search results are served from the index without checking for authorization, users can discover all *public content* that the search appliance has access to, regardless of whether they have authorization to view that content.

A user who is searching for protected content is prompted for her credentials once for each set of authentication mechanisms that share a username and password. The user enters her credentials on the **Universal Login Form**.

How a Search Appliance Labels Controlled-Access Content Sources as Public or Secure

When crawling and indexing controlled-access content over HTTP or HTTPS, the search appliance assigns public or secure status based on the type of crawl, and the **Make Public** checkbox in the Admin Console. If the **Make Public** checkbox is selected on the **Content Sources > Web Crawl > Secure Crawl > Forms Authentication** page, content is labeled as public. When the checkbox is cleared, content is labeled as secure.

Note that by selecting **Make Public**, all documents matching the URL pattern become public, even if ACLs are associated with the documents or the authmethod attribute in the feed record is set to a secure value.

The search appliance assigns status from these pages:

- Forms Authentication: Forms Authentication sites are controlled-access content sources that require the search appliance to obtain a session cookie from a login form. Most commercial single sign-on (SSO) solutions use this method of authentication. A search appliance can have multiple Forms Authentication rules for crawl and index. Forms authentication also configures actions for sites that require a session cookie to allow the search appliance to crawl the site.
- Web and content feeds: the `authmethod` attribute for the record specifies whether content is treated as public or secure.
  - To make feed content public, set the `authmethod` value to `none`. This is the default for content provided by feeds.
  - To make feed content secure, set the `authmethod` value to `ntlm`, `httpbasic`, or `httpsso`.

- Databases: All content from a database is labeled as public during serve.
- Connectors: If the connector supports authentication and authorization, and the **Make Public** checkbox is cleared, content from that connector is labeled as secure. In all other cases, content from a connector is labeled as public. To determine whether a connector instance supports authentication and authorization, look up Security Support in the Configuration guide for your connector.

In GSA release 7.4, the on-board connector manager and connectors are deprecated. They will be removed in a future release. If you have configured on-board connectors for your GSA, install and configure an off-board Google Connector. For more information, see the documentation that is available from the Connector Documentation page..

How a Search Appliance Determines What to Display in Public Search Results

The front end configuration for a search results page controls how much information users see for each item in the search results. When you make controlled-access content available for public search, open the **Page Layout Helper** or the **XSLT Stylesheet Editor** for each front end and review the stylesheet configuration to ensure that you are not revealing more information than the user needs.

In the **Page Layout Helper**, these parameters under **Search Results** control which information is displayed:

- When **Snippet** is selected, the `<S>` element is displayed in the search results. Clear the **Snippet** check box to remove snippets from the search results.
- When **Page Size** is selected, the `<C>` element's page size `SZ` value is displayed in the search results. Clear the **Page Size** check box to remove information about the document's size from the search results.
- When **Modified Date** is selected, the `<CACHE_LAST_MODIFIED>` element is included in the XML results. Clear the **Modified Date** check box to remove information about the document's freshness from the search results.
- When **Cache Link** is selected, the `<C>` element is included in the XML results. Clear the **Cache Link** check box to remove the link to the cached document from the search results.
- The **Result Page** navigation at the bottom of the page can indicate how many results are available. To prevent users from using this information to deduce how large your index is, choose the third option, which excludes both the "Goooooogle" navigation and the numbered references to search results pages.

In the **XSLT Stylesheet Editor**, these XSL variables control which information is displayed:

- `show_res_snippet` specifies whether to display a snippet for each result. Set `<xsl:variable name="show_res_snippet">0</xsl:variable>` to remove snippets from the search results.
- `show_meta_tags` specifies whether to display metadata for each result. Set `<xsl:variable name="show_meta_tags">0</xsl:variable>` to remove the document's metadata from the search results.
- `show_res_size` specifies whether to display the page size for each result. Set `<xsl:variable name="show_res_size">0</xsl:variable>` to remove information about the document's size from the search results.
- `show_res_date` specifies whether to display the last-modified date for each result. Set `<xsl:variable name="show_res_date">0</xsl:variable>` to remove information about the document's freshness from the search results.

- `show_res_cache` specifies whether to display the cache link for each result. Set `<xsl:variable name="show_res_cache">0</xsl:variable>` to remove the link to the cached document from the search results.
- `choose_bottom_navigation` specifies which navigation option to use at the bottom of the results page. Set `<xsl:variable name="choose_bottom_navigation">simple</xsl:variable>` to exclude both the "Goooooogle" navigation and the numbered references to search results pages.

Authentication

Serve-time authentication is the process of verifying the identity of a user who has issued a search request for controlled-access content. The Google Search Appliance uses these methods to establish the user's identity:

- Cookie-based authentication
- HTTP Basic or NTLM HTTP
- Kerberos authentication against a domain controller
- The SAML Authentication Service Provider Interface (SPI)
- LDAP
- Digital certificates and certification authorities

This section describes how a search appliance performs authentication, and how to configure authentication for the supported mechanisms. For information about how the search appliance determines whether an authenticated user, system, or service has access to secure content, see "Authorization" on page 41.

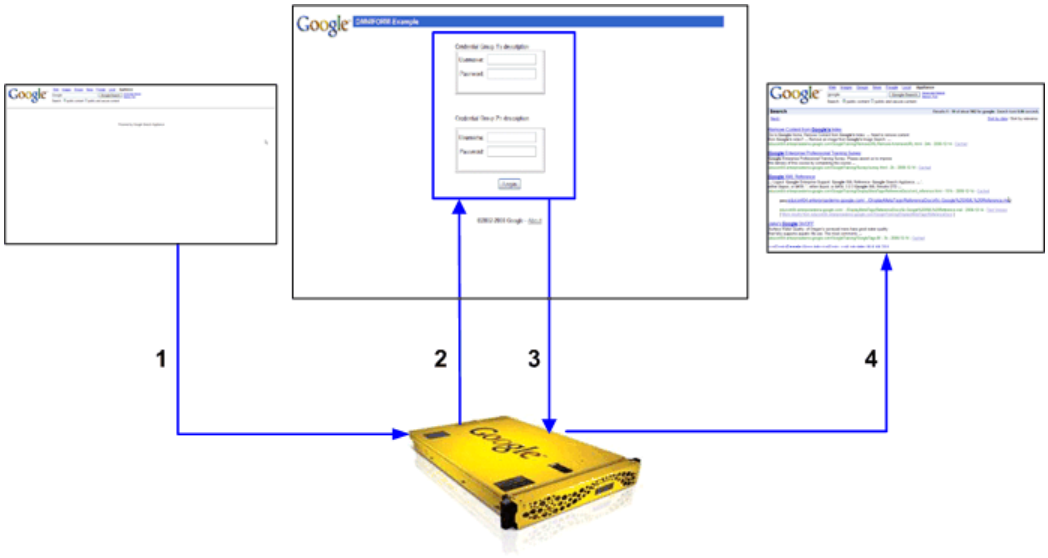
Universal Login

With Universal Login, a user who is searching for protected content is prompted for credentials once by the **Universal Login Form** for set of authentication mechanisms that share a username and password. The user is granted (or denied) access to the resources based on the credentials and the search appliance returns the appropriate search results. The Google Search Appliance supports Universal Login for the following authentication mechanisms:

- "Cookie-Based Authentication" on page 23 (single sign-on, forms)
- "HTTP-Based Authentication" on page 25 (HTTP Basic, NTLM)
- "Client Certificate-Based Authentication" on page 27
- "Kerberos-Based Authentication" on page 28
- SAML Authentication SPI (see"The SAML Authentication Service Provider Interface (SPI)" on page 34)
- "Connectors" on page 35
- "LDAP" on page 36

The Google Search Appliance also supports authentication without Universal Login using LDAP (see "Integrating the Search Appliance with an LDAP Server" on page 36).

The following diagram presents an overview of what happens when a user searches for protected content.



The numbers in the diagram refer to the following steps in the process:

1. The user performs a search against content in one or more protected resources.
2. The search appliance prompts the user once for all protected resources by using a single login page, the **Universal Login Form**.
3. The user enters one or more usernames and passwords for the protected resources on the **Universal Login Form** and submits it.

   The user is granted (or denied) access to the resources based on the credentials.
4. The search appliance returns search results, with denied resources filtered out.

Credential Groups

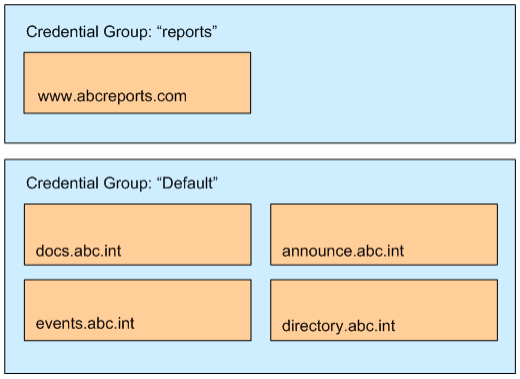
A credential group represents the set of authentication mechanisms that share a username and password. Credential groups enable the search appliance to gather user credentials by using the **Universal Login Form**.

For example, suppose the ABC company has the following basic authentication-based and forms authentication-based Single Sign-On (SSO) systems:

- `www.abcreports.com` uses forms authentication. This domain hosts business reports that are available for purchase.
- `documentation.abc.int` uses forms authentication. This domain hosts design documents for use by internal employees.
- `events.abc.int` uses HTTP Basic authentication. This domain contains information about internal company events
- `announce.abc.int` uses forms authentication. This domain contains announcements for employees.
- `directory.abc.int` uses forms authentication. This domain provides phone and office location information about employees.

The domain `www.abcreports.com` uses one, unique set of credentials (user name and password). All the other domains share a different single set of credentials.

Because ABC company's domains are protected by two sets of credentials, their search appliance administrator can group the domains into two credential groups, "reports" and "Default," as illustrated in the following diagram.



The search appliance prompts only for a single username/password for each credential group, and then attempts to verify it against the systems in the credential group.

A credential group can have any number of authentication mechanisms (also known as "credential group elements"). The search appliance supports any number of credential groups.

Currently, you can only add one domain protected by HTTP Basic authentication to the credential groups that you configure on a Google Search Appliance.

For information about setting up credential groups, see "Working with Credential Groups" on page 21.

Primary Verified Identity

Although the search appliance can track multiple verified user identities at once, it only currently supports one verified identity (primary verified identity) from any source, for example, when working with policy Access Control Lists (ACLs). The following list contains a list of mechanisms that can provide the primary verified identity, in order of precedence:

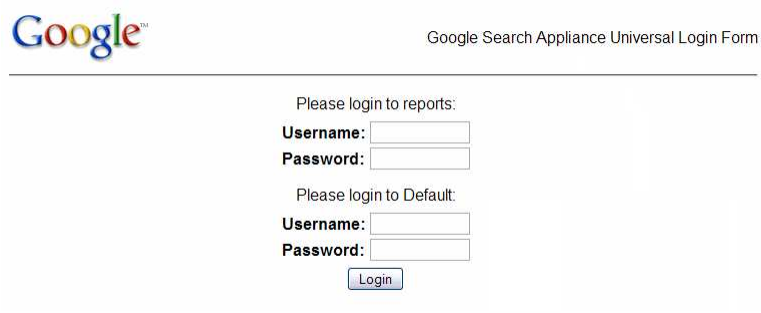
1. x.509 client certificate authentication
2. Universal Login Form—default credential group
3. Forms authentication
4. Kerberos
5. Basic authentication (only returns a verified identity when used with LDAP)
6. Connectors

In other words, a verified identity from x.509 client certificates overrides all other mechanisms, including a verified identity from the SAML, and so on.

Universal Login Form

After credential groups are configured, whenever a user performs a secure search, and the user is not already authenticated, the Google Search Appliance presents the **Universal Login Form**, shown in the following figure. The Universal Login Form is the primary way the search appliance gathers user credentials (usernames and passwords). The user's credentials are applied to all the systems in the credential groups for which the user supplies a username and password.

The **Universal Login Form** can contain multiple sets of user name and password fields—one set for each credential group.



You can use the default **Universal Login Form** or create one that is specific to your organization. For more information, see "Customizing the Universal Login Form" on page 49.

Credential Group Satisfaction

The following process gives an overview of how the **Universal Login Form** determines if a user's credentials satisfy configured credential groups:

1. The **Universal Login Form** checks the existing cookies that the user already has to see whether its configured credential groups are already satisfied. The authentication mechanism can return one of three answers: verified, rejected, or indeterminate (which usually means an error occurred and a definitive answer couldn't be found). If any mechanism answers "rejected," the credential group is not satisfied.
2. If all credential groups are satisfied, the **Universal Login Form** is skipped and appropriate results are displayed.

   If only some are satisfied, the logins for those credential groups are disabled (grayed-out).
3. The **Universal Login Form** presents a challenge for each configured-but-unsatisfied credential group.
4. The user enters her credentials for each unsatisfied credential group on the **Universal Login Form**.
5. The **Universal Login Form** attempts to verify each provided credential, and updates which credential groups are satisfied.
6. If any credential groups remain unsatisfied, the **Universal Login Form** is presented again (with only the unsatisfied credential group's enabled), up to three times.

Options that you, as the search appliance administrator, choose when configuring a credential group determine whether the user must enter credentials on the **Universal Login Form** to view search results. For more information about this topic, see "Creating Credential Groups" on page 21.

Working with Credential Groups

Set up credential groups by performing the following tasks in the Google Search Appliance Admin Console:

1. "Creating Credential Groups" on page 21
2. "Configuring Credential Groups" on page 22

It is important to configure a credential group once you create it. If there is an unconfigured credential group, the search appliance does not serve secure results. To avoid this issue, delete any unconfigured credential groups.

About the Default Credential Group

The Google Search Appliance provides a built-in credential group named **Default**. You can configure the **Default** credential group, as described in "Creating Credential Groups" on page 21 and "Configuring Credential Groups" on page 22. If you plan on using credential groups and policy ACLs, configure the **Default** credential group but do not rename it. For more information, see "Using Credential Groups with Policy ACLs" on page 48.

Creating Credential Groups

Create a new credential group by using the **Search > Secure Search > Universal Login** page in the search appliance Admin Console. For information about creating and maintaining credential groups, click **Admin Console Help > Search > Secure Search > Universal Login**.

For each credential group that you create, you can choose two options:

- **Require a User-name for this credential group?** (See "Require a User-Name Option" on page 21.)
- **Group is Optional?** (See "Group is Optional? Option" on page 22.)

The following sections describe these options.

Require a User-Name Option

The **Require a user-name for this credential group?** option ensures that the system has a username for an authenticated user. This option is important when your configuration uses cookie-based authentication in combination with an authorization mechanism that requires user-names, such as policy ACLs, SAML, and connectors.

If a user presents pre-existing cookies that are sufficient for access to configured sample URLs, but no cookie cracker is in use (see "Using Cookie Cracking" on page 40), the search appliance does not know the user's name. In this case, if the box is checked, the credential group is not pre-satisfied, even if the sample URL check succeeds, and a **Universal Login Form** is presented to the user. If a user-name is available, from a different authentication mechanism, a previous **Universal Login Form**, or a cookie cracker, then the group can be pre-satisfied, and if all credential groups are pre-satisfied, then the **Universal Login Form** is skipped altogether.

Group is Optional? Option

The **Group is optional?** option controls the behavior of the **Universal Login Form**.

If this option is checked, the user is not required to type a username and password in the **Universal Login Form** for this credential group. The user can submit the **Universal Login Form** and view search results. However, if the user does not login, then search results do not include secure results protected by that credential group.

If this option is not checked, the user is required to type a username and password in the **Universal Login Form**. The user cannot view any search results until he has supplied his username and password. He will keep being sent back to the **Universal Login Form** until he provides the correct credentials.

Adding a Credential Group

To create a new credential group:

1. Click **Search > Secure Search > Universal Login**.
2. In the **Credential Group Name** box, type a name for the new credential group.

   Credential group names can be up to 200 characters long and can contain only alphanumeric characters, underscores, and hyphens. A name cannot begin with a hyphen.
3. (Optional) Type the name that you want to appear on the **Universal Login** form in the **Credential Group Display Name** box. There are no character or format restrictions on the Credential Group Display Name.
4. Select **Require a user-name for this credential group?** and **Group is optional?**, as described in the preceding sections.
5. Click **Save**.

The new credential group's name appears in the list of credential groups.

Configuring Credential Groups

After you create a new credential group, you can configure it by adding credential group rules on the **Search > Secure Search > Universal Login Auth Mechanisms** page. This page provides tabs for a adding rules for the following types of authentication mechanisms:

- "Cookie-Based Authentication" on page 23
- HTTP-based or NTLM authentication (see "HTTP-Based Authentication" on page 25)
- "Client Certificate-Based Authentication" on page 27
- "Kerberos-Based Authentication" on page 28
- SAML (see "Configuring a Credential Group for SAML Authentication" on page 34)
- "Connectors" on page 35
- "LDAP" on page 36

Cookie-Based Authentication

During serve, secure content from sites that were crawled through a Forms Authentication rule are handled by cookie-based authentication.

Configuring a Credential Group for Cookie-Based Authentication

Configure a credential group rule for cookie-based authentication by supplying a URL pattern and sample URL on the **Search > Secure Search > Universal Login Auth Mechanisms > Cookie** page in the Admin Console. Optionally, you can also supply a redirect URL.

Sample URL

Supply a sample URL, which is any page in the protected site that all authenticated users can view. The sample URL is used to detect whether a user has correct credentials for a particular authentication method.

Each sample URL is checked before the **Universal Login Form** is presented, to determine if the user's initial set of cookies can "pre-satisfy" any or all credential groups. In additional, if any cookie-based authentication methods are defined, the search appliance uses credentials gathered in the **Universal Login Form** to gather cookies and then uses those cookies to retrieve the sample URL page. If the retrieval is successful, the credentials are verified as correct. If a user has the correct cookies, content is presented.

If a user does not have the correct cookies, the sample URLs page should redirect to the forms-based login system. To enable the sample URL to send a redirect response that leads to a login form, check **When sample URL fails, expect the sample page to redirect to a form, and log in to that form** on the **Search > Secure Search > Universal Login Auth Mechanisms > Cookie** page.

For the URL pattern `http://www.abcreports.com/`, an example of a sample URL is `http://www.abcreports.com/standard.html`.

Redirect URL

If you supply a redirect URL, the authentication mechanism changes significantly. In non-redirect mode, the search appliance transfers a username / password from the **Universal Login Form** to a login form found when attempting to retrieve the sample URL. With a redirect URL, the search appliance will automatically redirect to that URL. The service at that URL can then authenticate the user in whatever way it wishes. Upon completion of that authentication, the service at the redirect URL should grant a cookie to the user which provides access to secure content (and to the sample URL, if provided), and redirect the user back to the search appliance.

If a sample URL is provided, it allows the search appliance to skip the redirect if the user already has cookies that provide access to the sample URL. A sample URL also allows verification of the user cookies upon return from the sample URL service.

Possible advantages of redirect URL authentication:

- The user's password is never sent to the search appliance.
- The redirect URL server can interact directly with the user. This can facilitate login scenarios where the user's browser must perform operations (such as evaluating complex JavaScript) that the search appliance form-filling emulator cannot perform.

Disadvantages of redirect URL authentication:

- It is generally slower than standard cookie-based forms authentication.
- It requires setting up the server for the redirect URL to respect the return URL parameter, which gives the server for the redirect URL information about the quickest path back to the search appliance.
- It does not result in a verified user-name unless the sample URL is also a cookie cracker.

On balance, Google does not recommend using a redirect URL as a preferred method of authentication.

Adding a Credential Group Rule for Cookie-Based Authentication

To add a credential group rule for cookie-based authentication:

1. Click **Search > Secure Search > Universal Login Auth Mechanisms > Cookie**.
2. Select a credential group from the pull-down menu.
3. Optionally, click **When sample URL check fails, expect the sample page to redirect to a form and log in to that form**.
4. In the **Mechanism Name** box, type a unique name for the authentication mechanism. A mechanism name must not be the same as another mechanism name or credential group name. Mechanism names are case-sensitive and can be up to 200 characters long, and can contain only alphanumeric characters, underscores, and hyphens. A name cannot begin with a hyphen.
5. Type a sample URL for the site in the **Sample URL** box.
6. Optionally, type a URL in the **Redirect URL** box.
7. Optionally, type a **Return URL Parameter**.
8. Optionally, change the default time for the search appliance to make a network connection by entering the number of seconds in the **Timeout** box.
9. Optionally, type the number of seconds that the verification of user credentials will be trusted in the **Trust Duration** box.
10. Click **Save**.

For more information about how to configure a credential group for cookie-based authentication, click **Admin Console Help > Search > Secure Search > Universal Login Auth Mechanisms > Cookie**.

Multiple Cookie Domains

The Google Search Appliance can work with Cookie Provider of Computer Associates SiteMinder Web Access Manager in supporting multiple cookie domains.

For example, suppose your organization has the following two web servers hosted in different DNS domains:

- Web server A hosts Accounts.com
- Web server B hosts Investments.com

Authentication and authorization for web server A and web server B are controlled by disparate SiteMinder SSO servers. The Google Search Appliance is deployed in domain Accounts.com.

When a user performs a search against the Google Search Appliance, she provides her username and password to get access to the protected content. After the user is authenticated, SiteMinder Web Access Manager issues a set of session cookies that includes one cookie for the Accounts.com domain and another cookie for the Investments.com domain. In other words, the user logs in once, to Accounts.com, and through SiteMinder cross-domain single sign-on, she gains access to both Accounts.com and Investments.com.

The Google Search Appliance recognizes these correlated cookie domains and keeps the cookies synchronized.

Cookie-Based Authentication Scenarios

Different organizations set up cookie-based authentication rules for the Google Search Appliance's Universal Login in a variety of different ways. The selections that you, as a search appliance administrator, make by using the Admin Console depend on your system's capabilities and your organization's requirements. For examples of setting up cookie-based authentication, see "Cookie-Based Authentication Scenarios" on page 66.

HTTP-Based Authentication

During serve, secure content from sites that were crawled by using user accounts and passwords entered on the **Content Sources > Web Crawl > Secure Crawl > Crawler Access** page are handled by HTTP-based authentication.

Configuring a Credential Group for HTTP Basic or NTLM

Configure a credential group for HTTP-based authentication or NTLM by supplying a URL pattern and sample URL on the **Search > Secure Search > Universal Login Auth Mechanisms > HTTP** page in the Admin Console. To configure an authentication domain that is protected by NTLM instead of HTTP Basic, click the NTLM check box.

Sample URL

Supply a sample URL, which is any page in the protected site that all authenticated users can view. The sample URL is used to detect whether a user has correct credentials for a particular authentication method.

Each sample URL is checked before the **Universal Login Form** is presented, to determine if the user's initial set of cookies can "pre-satisfy" any or all credential groups. In additional, if any SSO Forms methods are defined, the search appliance uses credentials gathered in the **Universal Login Form** to gather cookies and then uses those cookies to retrieve the sample URL page. If the retrieval is successful, the credentials are verified as correct.

For the URL pattern `http://www.abcreports.com/`, an example of a sample URL is `http://www.abcreports.com/status.html`.

You can set up silent authentication with a sample URL page when the **Require a user-name for this credential group?** option is selected on the **Search > Secure Search > Universal Login** page by using cookie cracking. With silent authentication, users are authenticated without being directed to a login page. For information about this topic, see "Using Cookie Cracking" on page 40.

Adding a Credential Group Rule for HTTP Basic

To add a credential group rule for HTTP Basic authentication:

1. Click **Search > Secure Search > Universal Login Auth Mechanisms > HTTP**.
2. Select a credential group from the pull-down menu.
3. In the **Mechanism Name** box, type a unique name for the authentication mechanism. A mechanism name must not be the same as another mechanism name or credential group name. Mechanism names are case-sensitive and can be up to 200 characters long, and can contain only alphanumeric characters, underscores, and hyphens. A name cannot begin with a hyphen.
4. Type a sample URL for the site in the **Sample URL** box.
5. Optionally, change the default time for the search appliance to make a network connection by entering the number of seconds in the **Timeout** box.
6. Optionally, type the number of seconds that the verification of user credentials will be trusted in the **Trust Duration** box.
7. Click **Save**.

Adding a Credential Group Rule for NTLM

To add a credential group rule for NTLM authentication:

1. Click **Search > Secure Search > Universal Login Auth Mechanisms > HTTP**.
2. Select a credential group from the pull-down menu.
3. Click the **NTLM** check box.
4. In the **Mechanism Name** box, type a unique name for the authentication mechanism. A mechanism name must not be the same as another mechanism name or credential group name. Mechanism names are case-sensitive and can be up to 200 characters long, and can contain only alphanumeric characters, underscores, and hyphens. A name cannot begin with a hyphen.
5. Type a sample URL for the site in the **Sample URL** box.
6. Optionally, change the default time for the search appliance to make a network connection by entering the number of seconds in the **Timeout** box.
7. Optionally, type the number of seconds that the verification of user credentials will be trusted in the **Trust Duration** box.
8. Click **Save**.

For more information about how to configure a credential group for HTTP-based authentication or NTLM, click **Admin Console Help > Search > Secure Search > Universal Login Auth Mechanisms > HTTP**.

Client Certificate-Based Authentication

The search appliance can check a user's SSL certificate to verify that it was issued by a trusted certificate authority before serving secure results. This section provides a general overview of how to configure a search appliance to require X.509 Certificate Authentication from users who submit search queries.

Configure a search appliance for client certificate-based user authentication by performing the following tasks:

1. "Enabling User Authentication by X.509 Certificate" on page 27
2. "Configuring a Credential Group for Client Certificate-Based Authentication" on page 28

Enabling User Authentication by X.509 Certificate

To enable user authentication by X.509 certificate, the search appliance must have a digital certificate that permits crawl and serve over HTTPS. Also, client certificate authentication cannot be used for the head requestor, therefore configure policy ACLs (see "Policy Access Control Lists" on page 42) or the SAML authorization SPI (see "How to Exclude Controlled-Access Content Sources from Search" on page 48). The preloaded certificate authorities are enabled by default. You can disable them or re-enable them.

To configure the search appliance to require X.509 Certificate Authentication for search requests from users:

1. Log in to the Admin Console.
2. Choose **Administration** > **SSL Settings**. Configure the search appliance to permit crawl and serve over HTTPS by installing an SSL certificate. For details, see "Configuring Crawl and Serve Over HTTPS" on page 14.
3. On the **Administration > SSL Settings** page, check the settings for **Force secure connections when serving?**

   If **No** is selected, you must change it to one of the following options: **Use HTTPS when serving secure results, but not when serving public results** or **Use HTTPS when serving both public and secure results**.
4. Choose **Administration** > **Certificate Authorities**. Under **Add more Certificate Authorities**, enter the .pem file that contains your root CA certificate. The search appliance will trust certificates issued by this root certificate.
5. Choose **Administration** > **Certificate Authorities**. Under **Add Certificate Revocation List**, enter the file that contains the current certificate revocation list (CRL). The search appliance will NOT trust certificates that appear in this list. The CRL prevents a user with a revoked certificate from accessing secure content.
6. Optionally, to disable default certificate authorities, clear the **Enable default Certificate Authorities** checkbox under **Default Certificate Authorities**.
7. Click **Save**.

Configuring a Credential Group for Client Certificate-Based Authentication

To add a credential group rule for client certificate-based authentication to a credential group:

1. Click **Search > Secure Search > Universal Login Auth Mechanisms > Client Certificate**.
2. Select a credential group from the pull-down menu.
3. Click **Enable client certificate authentication support**.
4. In the **Mechanism Name** box, type a unique name for the authentication mechanism. A mechanism name must not be the same as another mechanism name or credential group name. Mechanism names are case-sensitive and can be up to 200 characters long, and can contain only alphanumeric characters, underscores, and hyphens. A name cannot begin with a hyphen.
5. Click **Save**.

Kerberos-Based Authentication

Kerberos is a network authentication protocol that enables client and server applications to perform mutual authentication for the duration of a user's login session. The search appliance can use Kerberos authentication by issuing a head request to confirm a user's right to view controlled-access documents. The search appliance only performs this check during secure serve for content on HTTP servers.

Kerberos supports the following encryption methods:

- rc4
- aes128-cts-hmac-sha1-96
- arcfour-hmac
- des3-cbc-sha1
- aes256-cts-hmac-sha1-96
- des-cbc-md5

To ensure that a search appliance uses Kerberos during serving, content sources must be enabled for Kerberos. For more information on ensuring that Kerberos is configured correctly on Windows content sources, see the wiki page http://code.google.com/p/google-saml-bridge-for-windows/wiki/ConfigKerberos (the information is provided as a reference, and is not officially supported by Google).

The Kerberos implementation supports:

- Windows IIS web sites with Kerberos enabled.
- Windows file share with Kerberos enabled.
- Linux/Unix file share using SMB in a Windows domain with a Windows AD as the Kerberos Key Distribution Center (KDC).
- Cross domain access.

Take note that the search appliance supports serving of SMB content via Kerberos only. It does not support crawling of SMB content via Kerberos.

With cross-domain access, the KDC associated with the search appliance can communicate with other KDCs to authenticate and authorize users from other domains. The secure content does not have to be in the same domain as the search appliance, but the two domains must have transitive trust enabled between them. For information about transitive trusts, see Microsoft documentation. In a Windows cross-domain configuration, the search appliance requires the DNS server to advertise KDCs for both domains by way of DNS SRV responses.

The Kerberos implementation does not support:

- Windows constrained delegation. Workaround: See Google SAML Bridge for Windows in *Enabling Windows Integrated Authentication*.
- Linux/Unix KDC.

When the search appliance is configured to use IWA / Kerberos authentication, the search appliance checks the user's session ticket against a KDC before displaying secure search results to a user. For Windows servers, the domain controller acts as the KDC for Kerberos authentication.

- If a user has a valid ticket, the user can see secure search results without having to log in again.
- The search appliance does not support NTLM fallback. If a user does not have a valid ticket, or is unable to perform Kerberos authentication against the search appliance, she might get prompted for credentials. However, the search appliance does not process those credentials. To configure NTLM fallback, use Google SAML Bridge for Windows, described in *Enabling Windows Integrated Authentication*.

To configure the search appliance to use IWA / Kerberos authentication:

1. Enroll the search appliance in the domain managed by your KDC (see "Enrolling the Search Appliance in the KDC Domain and Creating a Keytab File" on page 30). The KDC is typically a Microsoft Windows Server acting as a domain controller. As part of this step, you must also request and register a Kerberos key table, called a keytab file.
2. Log in to the Admin Console and configure a credential group for Kerberos (see "Configuring a Credential Group for Kerberos-Based Authentication" on page 31).
3. Ensure that your domain users have appropriate browser settings to use Kerberos authentication when querying the search appliance (see "Configuring Web Browsers for Kerberos Authentication" on page 32).

After you complete these steps, recrawl the affected content sources. The search appliance is then able to check a user's authentication status without requiring an additional login.

A verified identity from Kerberos authentication can be used for authorization. The following authorization mechanism can use the verified identity from Kerberos authentication:

- Policy ACLs
- SAML authorization SPI
- Connectors

If your content sources support these authorization mechanisms, then the content sources are not required to support Kerberos, and delegation is not required.

Enrolling the Search Appliance in the KDC Domain and Creating a Keytab File

The process for creating a user for your Key Distribution Center depends on the type of domain controller that you are using. This guide provides instructions for installing the search appliance on a Windows domain (RC4 and DES encryption).

Instructions for Microsoft Windows 2003, 2008, XP, and 7 (All Encryption types)

In the following instructions, you configure the search appliance as a user in Active Directory, then create a keytab file. The search appliance password in Active Directory must match the password in the keytab file.

To configure Windows:

1. Log into the Windows server that acts as the domain controller on your network.

   Note: The domain controller must have the latest Windows software version to ensure all supported encryption types are included.

2. Use the Active Directory Management wizard to create a new object-user account for the search appliance by entering the following information:

   - First Name and User Logon Name (the first name and login can be anything to help you identify the search appliance account. For example "gsa_account")
   - Password

3. Open the properties for the user. Use the **Account** tab for the search appliance account to modify and apply the following properties:

   a. Select the domain that you want to use from the drop-down box. Typically, there is only one domain listed.
   b. If the account does not use DES, clear the checkbox labeled **Use DES encryption types for this account**.
   c. Clear any other checkboxes under account properties.
   d. If permitted by your security policies, set **Password never expires**.

4. Open a command prompt.

5. At the command prompt, create a keytab file for the search appliance and register the search appliance as the principal by entering the following command:

   ```
   ktpass -princ HTTP/FQDN_of_the_searchappliance@DOMAIN_NAME
   -mapuser DOMAIN_NAME\searchappliance_username
   -pass searchappliance_password -out filename.keytab -ptype KRB5_NT_PRINCIPAL
   -crypto ALL
   ```

   where **FQDN**=fully qualified domain name.

Note: The search appliance username, password, and domain must be consistent with the user account that you created in step 2. With the exception of the `mapuser` switch, domain names must be fully qualified. Ensure that when you issue the `ktpass` command, HTTP is in upper-case letters and the string *FQDN_of_the_search_appliance* is in lower-case letters, as shown in the examples in this section. The *FQDN_of_the_search_appliance* must be the DNS A-name for the search appliance, not the CNAME. The *ptype* parameter specifies the principal type. The value must be *KRB5_NT_PRINCIPAL* (general ptype)**.**

For example, suppose the domain is FOODOMAIN, the user account is gsa_account, the user password is 123pass, and the FQDN of the search appliance is gsa.foodomain.com.

You would enter the following command:

```
ktpass -princ HTTP/gsa.foodomain.com@FOODOMAIN.COM
-mapuser FOODOMAIN\gsa_account -pass 123pass -out myfilename.keytab
-ptype KRB5_NT_PRINCIPAL -crypto ALL
```

The keytab file is the Kerberos key table that you will install on the search appliance.

6. At the command prompt, enter the following command, where `demo.keytab` is the keytab name. This lists encryption types so you can verify that all required types are included:

   ```
   klist -ke demo.keytab
   ```

   Encryption types are shown in ( ) in the command output:

   ```
   KVNO Principal
   ---- --------------------------------------------------------------------------
     3 HTTP/search.mydomain.com@MYDOMAIN.COM (DES cbc mode with CRC-32)
     3 HTTP/search.mydomain.com@MYDOMAIN.COM (DES cbc mode with RSA-MD5)
     3 HTTP/search.mydomain.com@MYDOMAIN.COM (ArcFour with HMAC/md5)
     3 HTTP/search.mydomain.com@MYDOMAIN.COM (AES-256 CTS mode with 96-bit SHA-1
   HMAC)
     3 HTTP/search.mydomain.com@MYDOMAIN.COM (AES-128 CTS mode with 96-bit SHA-1
   HMAC)
   ```

7. If Kerberos will be used for authorization, open the search appliance user account properties again. On the **Delegation** tab of **User properties**, select **Trust this user for delegation to any service**.

   If you want to use Kerberos for authentication only and use another service, such as policy ACLs, SAML authorization SPI, or connectors, for authorization, then you do not have to enable delegation.

8. On the **Account** tab of **User properties**, verify that the user logon name field was populated with the HTTP/ prefix, for example, `HTTP/`*`FQDN_of_the_search_appliance`*.

Configuring a Credential Group for Kerberos-Based Authentication

To configure Kerberos-based authentication in the Admin Console:

1. On the server where you created the keytab file, open a web browser and log into the Admin Console on the search appliance.
2. Choose **Search > Secure Search > Universal Login Auth Mechanisms > Kerberos**.
3. Under **Specify a Kerberos Key Distribution Center (KDC)/Windows Domain Controller (DC)**, type the KDC host domain name in the **Kerberos KDC Hostname** box.
4. Optionally, to enable cross-domain access, click the **Enable KDC DNS Lookup** checkbox.
5. Click the **Save Kerberos KDC Hostname** button.

6. Under **Import a Kerberos Service Key Table ("keytab") File**, type the path name for the keytab file in the **Keytab File Name** box or click **Browse** to navigate to the file.
7. Click the **Import Kerberos Keytab File** button.
8. Select a credential group from the pull-down menu.
9. Click the **Enable Kerberos support** checkbox.
10. In the **Mechanism Name** box, type a unique name for the authentication mechanism. A mechanism name must not be the same as another mechanism name or credential group name. Mechanism names are case-sensitive and can be up to 200 characters long, and can contain only alphanumeric characters, underscores, and hyphens. A name cannot begin with a hyphen.
11. If the KDC is using single-DES encryption, click **Allow Weak Crypto**.

    If you do not check this box and you try to enable Kerberos-based authentication with a KDC using single-DES encryption, an error message appears.
12. Click **Save**.

For more information about how to configure Kerberos based authentication, click **Admin Console Help > Search > Secure Search > Universal Login Auth Mechanisms > Kerberos**.

Configuring Web Browsers for Kerberos Authentication

Users who query the search appliance must have their web browsers configured to use Kerberos authentication.

Safari is not a supported browser because it does not forward Kerberos tickets. You can find more information about this issue at http://openradar.appspot.com/6644527.

Configuring Internet Explorer

To configure Internet Explorer:

1. Start Internet Explorer and select **Tools** > **Internet Options**.
2. The search appliance URL must be defined in the **Local Intranet** zone or the **Trusted Sites** zone. If the search appliance is already part of the **Trusted** or **Intranet** zones, you can skip this step.
   a. On the **Security** tab, select the **Local Intranet** web zone, and click the **Sites**... button.
   b. In the **Local intranet** dialog, click the **Advanced** button.
   c. Under **Add this Web site to the zone**, enter the search appliance's URL and click **Add**.
   d. Leave the **Require server verification (https:) for all sites in this zone** setting as it is. This option controls whether communication with the search appliance requires SSL certificates. For more on certificate use, see "Configuring Crawl and Serve Over HTTPS" on page 14.
   e. Click the **OK** button, then click **OK** again to save this change and return to **Internet Options**.
   f. With **Local Intranet** zone selected, click the **Custom level** ... button and verify that **Automatic logon only in Intranet zone** is checked.

      If you cannot include the search appliance in the **Local Intranet** zone, add it to the **Trusted Sites** zone and select **Automatic logon with current user and password**.
3. Choose the **Advanced** tab.
4. Under **Security**, select the checkbox labeled **Enable Integrated Windows Authentication (requires restart)**. This sets the browser to use Kerberos authentication.
5. Click **OK** and restart Internet Explorer.

Configuring Firefox/Mozilla

To configure Firefox/Mozilla:

1. Start Firefox.
2. In the address bar at the top of the window, enter the command "`about:config`".
3. Double-click `network.negotiate-auth.trusted-uris`. Modify this parameter to include the search appliance's URL as a trusted URI.
4. Double-click `network.negotiate-auth.delegation-uris`. Modify this parameter to include the search appliance's URL as a delegation URI.
5. If you are using a Microsoft Windows domain controller and you are running Mozilla Firefox on Microsoft Windows, verify that `network.auth.use-sspi` is set to true, which is its default value.

Note: For more on Mozilla and integrated authentication, see http://www.mozilla.org/projects/netlib/integrated-auth.html.

Configuring Google Chrome

If Google Chrome is running on Windows and the GSA is in the local Intranet zone, configuration is not required for Kerberos. Select **Local Intranet zone**, click the **Custom level...** button and verify that **Automatic logon only in Intranet zone** is checked.

To configure Google Chrome in a non-Windows environment:

1. Confirm Chrome is at build 7.0.518.0 or above.
2. Start Chrome with the following command line switches:
   - --auth-server-whitelist=[search appliance hostname] to include the search appliance's hostname as a trusted URI
   - --auth-negotiate-delegate-whitelist=[search appliance hostname] to include the search appliance's hostname as a delegation URI

Note: For more information on Chrome command line switches, see: http://www.chromium.org/developers/how-tos/run-chromium-with-flags. For more information on Authentication in Chrome, see: http://dev.chromium.org/developers/design-documents/http-authentication.

More Kerberos Information

For more information about the Google Search Appliance and Kerberos, see the following documents:

- *How the Google Search Appliance uses Kerberos to Authenticate Users and to Authorize Users to See Content* (http://support.google.com/gsa/answer/6055202)
- *Troubleshooting Kerberos secure searches* (http://support.google.com/gsa/answer/6055171)
- *Google Search Appliance IWA/Kerberos test client* (http://code.google.com/p/gsa-kerberos-test-client/)

The SAML Authentication Service Provider Interface (SPI)

The Authentication and Authorization Service Provider Interfaces (SPIs) enable a search appliance to communicate with an existing access control infrastructure using standard SAML messages.

This section describes the Authentication SPI. For information about the Authorization SPI, see "How to Exclude Controlled-Access Content Sources from Search" on page 48. For more detailed information about how the Authentication and Authorization SPIs work, see the *Authentication/Authorization for Enterprise SPI Guide*.

When implemented, the Authentication SPI allows search users to authenticate to the search appliance. It is designed to allow customers to integrate the search appliance into an existing access control infrastructure. Instead of authenticating search users itself, the search appliance redirects the user to an Identity Provider (IP), a customer-implemented server, where the actual authentication takes place. The IP then redirects the user back to the search appliance, while passing information that includes the identity of the search user.

Before using the Authentication and Authorization SPI, you must configure the appliance to crawl and index some secure controlled-access content. For more information, see "Configuring Crawl for the SAML Authentication and Authorization Service Provider Interface" on page 13. The SPIs are only used when a user queries for secure results.

Configuring a Credential Group for SAML Authentication

When the Google Search Appliance is configured with a credential group that includes a SAML authentication domain, a user performing a secure search is challenged by the SAML Identity Provider. The user provides her credentials on the Identity Provider login page.

You can add a rule for SAML authentication to a credential group by specifying the Entity ID and login URL of the Identity Provider on the **Search > Secure Search > Universal Login Auth Mechanisms > SAML** page in the Admin Console. Using this page, you can also specify the binding in which the search appliance communicates with the SAML server:

- HTTP Artifact binding—To specify HTTP Artifact binding, enter an **Artifact Resolver URL**
- HTTP POST binding—To specify HTTP POST binding, enter the **Public Key of IDP**

You must specify either the Public Key of IDP or an Artifact Resolver URL in a credential group rule for SAML, but do not specify both.

When creating credential groups for the authentication mechanism, ensure that **Requires a User-Name** is selected. For more information, see "Require a User-Name Option" on page 21.

Artifact Resolver URL

The artifact resolver URL is the URL for the server that converts a returned artifact into a response message. If you provide the Artifact Resolver URL, the SAML server returns its responses using HTTP Artifact binding. If you specify an Artifact Resolver URL, do not specify an Identity Provider public key.

Public Key of IDP

The Identity Provider public key is used for signing an assertion. If you specify a public key, the search appliance tries to verify the digital signature of the assertion and the SAML server returns its responses using HTTP POST binding. If you specify an Identity Provider public key, do not specify an Artifact Resolver URL.

Adding a Credential Group Rule for SAML Authentication

If there are additional credential groups besides the one with the SAML entry, the search appliance challenges the user with the **Universal Login Form**. After the user provides her credentials on the **Universal Login Form**, the search appliance combines the verified identities from SAML and the **Universal Login Form**. The user is granted access to the resources based on the combined credentials.

To add a credential group rule for SAML authentication to a credential group:

1. Click **Search > Secure Search > Universal Login Auth Mechanisms > SAML**
2. Select a credential group from the pull-down menu.
3. In the **Mechanism Name** box, type a unique name for the authentication mechanism. A mechanism name must not be the same as another mechanism name or credential group name. Mechanism names are case-sensitive and can be up to 200 characters long, and can contain only alphanumeric characters, underscores, and hyphens. A name cannot begin with a hyphen.
4. Provide the IDP Entity ID and Login URL
5. Provide either an artifact resolver URL or a public key of IDP, but not both.
6. Click **Save**.

For more information about how to add a rule for SAML authentication to a credential group, click **Admin Console Help > Search > Secure Search > Universal Login Auth Mechanisms > SAML**.

Connectors

You can configure an authentication domain for a connector instance with support for the authentication Service Provider Interface (SPI).

Configuring a Credential Group for a Connector

To add a credential group rule for a connector instance:

1. Click **Search > Secure Search > Universal Login Auth Mechanisms > Connectors**.
2. Select a credential group from the pull-down menu.
3. In the **Mechanism Name** box, type a unique name for the authentication mechanism. A mechanism name must not be the same as another mechanism name or credential group name. Mechanism names are case-sensitive and can be up to 200 characters long, and can contain only alphanumeric characters, underscores, and hyphens. A name cannot begin with a hyphen.
4. Select a connector name from the pull-down menu.
5. Optionally, if you want the connector to lookup a user's group information without performing authentication, check **Perform group lookup only**.

   If you want the connector to lookup group information and perform authentication, leave the checkbox unchecked.
6. Optionally, change the default time for the search appliance to make a network connection by entering the number of seconds in the **Timeout** box.
7. Optionally, type the number of seconds that the verification of user credentials will be trusted in the **Trust Duration** box.
8. Click **Save**.

For more information about how to configure an authentication domain for a registered connector instance, click **Admin Console Help > Search > Secure Search > Universal Login Auth Mechanisms > Connectors**.

For comprehensive information about connectors, see documentation for the Google Search Appliance connectors (http://support.google.com/gsa/answer/2731901).

LDAP

For a search appliance to use LDAP for user authentication at serve time, you must perform the following tasks:

1. Integrating the search appliance with and LDAP server, as described in the following section.
2. Enabling LDAP authentication for the search appliance, as described on "Enabling LDAP Authentication for a Search Appliance" on page 38.
3. Enabling group lookup, as described on "Enabling Group Lookup" on page 38.
4. Configure a credential group rule for LDAP, as described in "Configuring a Credential Group for LDAP" on page 38.
5. Protecting the user's credentials for serve with HTTP Basic and NTLM HTTP, as described on "Protecting the User's Credentials for Serve with HTTP Basic and NTLM HTTP" on page 39.

Integrating the Search Appliance with an LDAP Server

If you are not using Kerberos authentication, and want to enable the search appliance to validate a user's login name and password by using a Lightweight Directory Access Protocol (LDAP) server, enable Directory Integration. This section provides a general overview of how to enable the search appliance to authenticate credentials against one or more LDAP servers. When a user connects to the Google Search Appliance and requests a search for secure results, the search appliance asks for credentials from the user. These credentials are then forwarded to an LDAP server for validation.

Note: The search appliance does not support using LDAP and Kerberos authentication at the same time; you must choose one method for all servers on your domain.

To specify LDAP settings for the search appliance:

1. Log in to the Admin Console.
2. Choose **Administration > LDAP Setup**.
3. Click **Create new LDAP Server**. The LDAP setup options appear.
4. In the **LDAP Directory Server Address** section, enter the following information:
   - **Host**—LDAP directory server's host name, which is a fully-qualified domain name or an IPv4 address.
   - **Port number** (optional)—the port number where the LDAP server listens for requests.
5. If your LDAP server does not allow anonymous users to search, enter the following user credentials that the search appliance uses when logging into the LDAP server:
   - **Distinguished Name (DN)**—A login on the LDAP server to which the search appliance connects to send authentication requests. If the LDAP server supports anonymous binds (authentication requests), you do not need to specify a DN.
   - **Password** (optional)—The password for the DN.

6. (Optional) Click the **Go to advanced settings page even if detection fails** checkbox.
7. Click **Continue**.

   The search appliance attempts to auto-detect the settings of the **LDAP Search Base**, the **User Search Filter**, the **Group Search Filter**, and if **SSL Support** exists and displays what it has detected. The advanced settings appear. If you have any version of Active Directory, the resolve nested groups operator (:1.2.840.113556.1.4.1941:) is automatically populated in **Group Search Filter**. Nested group lookup is not supported for Windows 2003 SP1 or older. To use group lookup for Active Directory running on Windows 2003 SP1 or older, you must remove the resolve nested groups operator.
8. If the LDAP server is used to authenticate administrators to the search appliance, specify the LDAP groups against which they will be authenticated:
   - **Superuser Group**—Any member of this group is considered an Admin Console administrator.
   - **Manager Group**—Any member of this group is considered an Admin Console manager.
9. Test the LDAP server settings for a potential search user by entering the following information in the **LDAP Search User Authentication Test** box and clicking **Test LDAP Settings**:
   - **Username**—The user name that enables the search appliance to connect to the LDAP server (relative to the search base)
   - **Password**—The password for the user name that enables the search appliance to connect to the LDAP server

   If the LDAP authentication succeeds, a listing appears similar to (in a Unix or Posix environment—Windows LDAP servers have a different format):

   ```
   uid - (user ID)
   ou - (organizational unit)
   dc - (company name)
   ```

   If the **LDAP Authentication Test** settings do not successfully authenticate the user, click **Cancel**, revisit and change the information you entered, and test again.
10. Test the LDAP server settings for administrator authentication by entering the following information in the **LDAP Administrator Authentication Test** box and clicking **Test LDAP Settings**:
    - **Username**—The administrator user name that enables the search appliance to connect to the LDAP server. (Relative to the search base.) To authenticate, the administrator must be a member of the LDAP Manager Group.
    - **Password**—The password for the administrator user name.

    If the LDAP authentication succeeds, a listing appears similar to (in a Unix or Posix environment—Windows LDAP servers have a different format):

    ```
    uid - (user ID)
    ou - (organizational unit)
    dc - (company name)
    ```

    If the **LDAP Authentication Test** settings do not successfully authenticate the administrator, click **Cancel**, revisit and change the information you entered, and test again.
11. After the LDAP Authentication Test is successful, click **Save LDAP Settings**.

    The search appliance has an internal memory authorization cache to avoid wasting bandwidth and time verifying the same credentials multiple times. The cache remains for an hour by default.
12. Open a search page in a browser or click **Test Center**, click **public and secure content**, and perform a search against the search appliance.

13. At the authentication prompt, perform the following test:
    a. Enter a user name with the wrong password. The authentication prompt reappears and prompts again.
    b. Enter the correct information to see the requested search results.

Enabling LDAP Authentication for a Search Appliance

To enable LDAP on a search appliance, click the **Enable Authentication** checkbox on the **Search > Secure Search > Universal Login Auth Mechanisms > LDAP** page. For more information, see "Configuring a Credential Group for LDAP" on page 38.

Enabling Group Lookup

You can enable a search appliance to automatically look up group information for a user during authentication, provided that the search appliance has a verified identity for the user.

To look up group information for a user, the search appliance uses the combination of group information from all its available sources. For example, if the search appliance has group information in the form of policy ACLs, it looks up group information for the user in the policy ACLs.

Group lookup works only if LDAP is correctly configured for the search appliance. However, group lookup works even if LDAP is not enabled for the search appliance.

Nested group lookup is supported for Windows 2003 SP2 and later only. To use group lookup for Active Directory running on Windows 2003 SP1 or older, you must remove the resolve nested groups operator (:1.2.840.113556.1.4.1941:) after it has been populated in **Group Search Filter**.

To enable group lookup, click the **Enable group lookup** checkbox on the **Search > Secure Search > Universal Login Auth Mechanisms > LDAP** page. For more information, see "Configuring a Credential Group for LDAP" on page 38.

Configuring a Credential Group for LDAP

To add a credential group rule, enable LDAP and automatic lookup of group information:

1. Click **Search > Secure Search > Universal Login Auth Mechanisms > LDAP**.
2. Select a credential group from the pull-down menu.
3. In the **Mechanism Name** box, type a unique name for the authentication mechanism. A mechanism name must not be the same as another mechanism name or credential group name. Mechanism names are case-insensitive and can be up to 200 characters long, and can contain only alphanumeric characters, underscores, and hyphens. A name cannot begin with a hyphen.
4. Select an LDAP configuration name from the pull-down menu.
5. Click the **Enable Authentication** checkbox.
6. Click the **Enable group lookup** checkbox.
7. Optionally, change the default time for the search appliance to make a network connection by entering the number of seconds in the **Timeout** box.
8. Optionally, type the number of seconds that the verification of user credentials will be trusted in the **Trust Duration** box.
9. Click **Save**.

Protecting the User's Credentials for Serve with HTTP Basic and NTLM HTTP

When a user performs a query for secure content, the search appliance responds with the same protocol. Because the responses for serve over HTTP Basic and NTLM HTTP include authorization headers, a malicious user could intercept the message and extract the header. To protect the user's credentials against such an attack, you can force the use of HTTPS during serve, even when the search request is sent over HTTP.

To specify whether the search appliance serves all content over HTTPS:

1. Log in to the Admin Console.
2. Choose **Administration** > **SSL Settings**. Scroll down to **Force secure connections when serving?**.
   - To return results in the protocol used by the original search query, choose **No**. This option is the least secure.
   - To force the search appliance to use HTTPS for secure content only, choose **Use HTTPS when serving secure results, but not when serving public results**.
   - To force the search appliance to use HTTPS for all content, choose **Use HTTPS when serving both public and secure results**. This option is the most secure.
3. Click **Save**.

Using Silent Authentication

With silent authentication, users are authenticated without being directed to a login page. The following table lists methods that provide silent authentication. Some methods produce a primary verified identity (see "Primary Verified Identity" on page 19). Because a primary verified identity is required for policy ACLs (see "Policy Access Control Lists" on page 42), these methods can be used with them. The methods that do not produce a primary verified identity cannot be used with policy ACLs, SAML, or connectors.

Method	Description
Cookie-based	If **Require a user-name?** (see "Require a User-Name Option" on page 21) is not checked, inbound cookie forwarding provides silent authentication without a primary verified identity; it cannot be used with policy ACLs, SAML, or connectors. If **Require a user-name?** (see "Require a User-Name Option" on page 21) is checked, then silent authentication can only be achieved with cookie cracking (see "Using Cookie Cracking" on page 40).
Kerberos	Authentication is always silent, produces a verified identity, group data can only from the internal GData-based database.
SAML	Can be silent, depending on how the SAML server is configured, produces a primary verified identity, can return group data.
x.509 https client certificate	Authentication is always silent, always produces a primary verified identity, group data can only from the internal GData-based database.

Using Cookie Cracking

If a credential group requires a user name for an authenticated user, you can implement silent authentication for content in the credential group (see “Credential Groups” on page 18) by using cookie cracking. Google recommends using the Require a user-name? option (see “Require a User-Name Option” on page 21) when you are using policy ACLs, authorization results caching, SAML authorization, or connector authorization. When you use cookie cracking, inbound cookie forwarding (from the content server to the search appliance) provides a username and or group.

To implement cookie cracking, if a sample URL check for user credentials is successful, the web server that runs the sample URL page generates the following response HTTP header (in addition to the standard headers):

```
X-Username:value
X-Groups: value1, value2
```

where *value* becomes a verified identity for the credential group that is associated with the sample URL.

The effect of the response header is that it has “cracked” open the cookie and revealed the user and/or group name. The cookie can be used to “pre-satisfy” the credential group and the user has access to protected content without having to re-enter his credentials.

Other than setting up a sample URL, there is no configuration required for using cookie cracking on a search appliance. However, to use cookie cracking, the content server administrator must modify the content server so that it returns the appropriate response header.

Note that the content server must only emit the X-Username and X-Groups headers when it is presented with a valid cookie. If the content server produces something like “X-Username: invalid cookie,” then all users with invalid cookies obtain “invalid cookie” as a verified identity, which could cause authorization caching to provide the incorrect results to some users.

There is a 3 second timeout limit for checking the sample URL. If the response time of the host is beyond this limit, the check for user credentials is not successful.

Using Perimeter Security

Perimeter security ensures that the search appliance doesn’t serve any results without user authentication.

When perimeter security is enabled, the search appliance prompts the user for credentials when he first submits a search request. The search appliance authenticates the user by using the mechanisms that are configured for Universal Login.

If the user is successfully authenticated, the search appliance serves results. If the user is searching for public content only, no authorization is required to view results. If the user is searching for both public and secure content, the search appliance uses the credentials it has gathered to perform authorization on secure documents. The user is not prompted again for credentials.

If the user cannot be authenticated, the search appliance doesn’t serve any results.

To configure perimeter security, use the **Search > Secure Search > Universal Login** page. For instructions for configuring perimeter security, click **Admin Console Help > Search > Secure Search > Universal Login**.

Authorization

Authorization is the process that determines whether an authenticated user, system, or service has permission to perform a task. After the search appliance authenticates a user by establishing the user's identity, the search appliance attempts to determine whether a user has access to the secure content that matches their search.

Flexible Authorization

Flexible authorization gives you control over authorization by enabling you to:

- Specify authorization mechanisms in your environment
- Customize which authorization mechanisms handle which URLs

You can perform these tasks by configuring flexible authorization rules. A flexible authorization rule defines:

- The protected content to which the rule applies
- An identity that maps the rule to a credential group or instance of an authentication mechanism
- Information that is specific to the authorization mechanism

You can configure rules for the following authorization mechanisms:

- CACHE
- CONNECTOR
- DENY
- HEADREQUEST
- POLICY
- SAML
- PER-URL ACL
- FILE_SYSTEM (SMB URLs)

To configure rules for authorization mechanisms, use the **Search > Secure Search > Flexible Authorization** page. For step-by-step procedures for configuring specific types of rules, click **Admin Console Help > Search > Secure Search > Flexible Authorization**.

After the search appliance authenticates a user by establishing the user's identity, the search appliance attempts to determine whether a user has access to the secure content that matches her search. The search appliance performs authorization checks by applying flexible authorization rules in the order in which they appear on the **Search > Secure Search > Flexible Authorization** page.

Although you can configure the authorization routing table, Google recommends using the default setting where the first rule in the table is for PER-URL ACLs. This setting provides the best authorization performance for a larger number of documents. Changing the order of the authorization rules in the table so that a rule for another mechanism is first might lead to slow authorization performance for a smaller number of documents. Google recommends always using the PER_URL_ACL mechanism with pattern "/" as the first rule, with or without late binding.

Most of the supported authorization mechanisms are capable of returning one of three possible decisions for each URL:

- Allow—Allow the user access to the URL.
- Deny—Deny the user access to the URL.
- Indeterminate—A definitive answer could not be determined, so the search appliance applies the following rule in the ordered list of rules.

Any given URL might match more than one flexible authorization rule. In this instance, each associated mechanism in the list is applied in order until one of them returns a decision other than indeterminate. If all mechanisms return indeterminate, or no mechanisms match, then the user is denied access to the URL. If a mechanism cannot handle a URL, it returns a decision of indeterminate.

Policy Access Control Lists

A policy ACL (Access Control List) provides information to the search appliance about which users or groups have access to a specific URL. By specifying policy ACLs on a search appliance, you can enhance performance and reduce load. Policy ACLs speed up the process of authorization and reduce the load on the authorization servers that occurs from performing HEAD requests to a remote authorization server.

Policy ACLs typically store the results that would have occurred if the search appliance initiated a HEAD request to verify authorization. However policy ACLs can also be used to override the decision that would have been returned by a HEAD request. For example, if you put in a policy ACL rule that permits a group to see all documents at a URL, but at the source repository (that is, the HEAD request), there's a more fine-grained rule where only some in the group can view documents, then the behavior with the policy ACL rule is that everyone can see the search results, but only those who have access rights can click the links.

Policy ACLs require that you use an authentication method to establish the identity of the user or group that you specify in the Policy ACL rules. You must specify domain name in the policy ACL. The domain format depends of the authentication method used by the search appliance.

For more information on policy ACLs, see the *Policy ACL API Developer's Guide*.

Per-URL ACLs and Policy ACLs

The search appliance supports two types of access control lists:

- Per-URL ACL—An ACL in the index that is associated with a single URL. A per-URL ACL has a limit of 100,000 entries (users and groups). ACL information can be applied to groups of documents through inheritance. For more information, see "Per-URL ACLs and ACL Inheritance," in the *Feeds Protocol Developer's Guide*.
- Policy ACL—A URL-pattern rule-based ACL. This type of ACL can have many URLs associated with it. For more information, see "Policy ACLs" on page 44.

Occasionally, a URL can be associated with both types of access control lists. You can choose which type takes precedence, as described in "Flexible Authorization" on page 41.

Methods for Adding ACLs to the Index

The search appliance supports different methods for adding per-URL ACLs and policy ACLs to the index. The following table lists these methods and provides references to documentation for each method.

ACL Type	Method	Comments	Described In
Per-URL ACL	Feed	Use a feed to push per-URL ACLs to the search appliance.	"Per-URL ACLs and ACL Inheritance" in *Feeds Protocol Developer's Guide*
	Connector	Use a connector to push per-URL ACLs to the search appliance (uses feeds).	
	Crawl document header	At crawl time, add per-URL ACLs, along with documents, through the X-GSA-External-Metadata HTTP response header.	"Crawling Per-URL ACLs" on page 43
	Specify in metadata (deprecated)	Define per-URL ACL can be defined in external metadata or metadata in the document itself.	"Legacy Metadata Format (Deprecated)" in *Feeds Protocol Developer's Guide*
Policy ACL	**Search > Secure Search > Policy ACLs** page in the Admin Console	Specify rules or import a text file that contains policy ACL rules.	"Policy ACLs" on page 44 and **Admin Console Help > Search > Secure Search > Policy ACLs**
	Google Search Appliance Policy ACL API.	Programmatically add policy ACL rules.	*Policy ACL API Developer's Guide*

Crawling Per-URL ACLs

At crawl time, the search appliance can accept per-URL ACLs, along with documents, through the `X-GSA-External-Metadata` HTTP response header. To include a per-URL ACL, specify the names of the groups or users that have access. The metadata supplied at crawl time replaces any previously indexed metadata. Per-URL ACLs contained in an HTTP header are considered external metadata and will empty and replace any metadata from feeds. For more information about external metadata in HTTP headers, see External Metadata Sent in an HTTP Header in the *External Metadata Indexing Guide*.

Note: Crawled content with per-URL ACLs will serve only for the "Default" credential group configured in Universal Login Auth Mechanisms but not for other credential groups.

To use this method of indexing per-URL ACLs, the web service that stores the content needs to be designed to generate the optional *X-GSA-External-Metadata* HTTP header. The header includes a comma separated list of encoded values in the following format:

`X-GSA-External-Metadata`: *value_1, value_2,...*

Where each value has the form *meta-name=meta-value*.

To specify a group, replace *meta-name* with `google:aclgroups` and *meta-value* with a single group name. For example, to specify engineering ("eng") as the group that has access to the URL, use `google:aclgroups=eng`.

To specify a user, replace *meta-name* with `google:aclusers` and *meta-value* with a single user name. For example, to specify Maria as the user that has access to the URL, use `google:aclusers=Maria`.

Both the *meta-name* and the *meta-value* are encoded according to section 2 of RFC3986 (http://www.ietf.org/rfc/rfc3986.txt) (commonly known as percent-encoding). The following example shows an encoded header:

```
X-GSA-External-Metadata: google%3Aaclusers=Maria, google%3Aaclgroups=eng
```

The per-URL ACLs supplied at crawl time are added to the search appliance index, replacing previously indexed per-URL ACLs. Subsequently crawled per-URL ACLs replace the previously indexed ones. If no external metadata header is supplied, the per-URL ACL in the index remains unchanged.

Any per-URL ACLs that are added later using a metadata-and-url feed are not merged with the crawled per-URL ACLs. An empty metadata-and-url feed clears all previous per-URL ACLs.

Policy ACLs

A policy ACL is expressed as a rule based on URL patterns. A policy ACL rule has two parts:

- URL Pattern to Protect (see "URL Pattern to Protect")—A URL pattern that you want to protect with restricted access.
- Allowed Users or Groups (see "Allowed Users or Groups" on page 45)—Lists the users or groups that have access to the restricted URL.

For example, suppose the eng (engineering) group is the only group that you permit to view all documents in the `example.com/engsite` page. To grant the engineering group access to the `engsite` page, specify a policy ACL rule:

```
example.com/engsite group:eng
```

When a search appliance executes a search, it attempts to match URLs that the search appliance retrieves from the index against policy ACLs. If a URL pattern matches the policy ACL rule, the search appliance applies the rule.

URL Pattern to Protect

You can specify a URL pattern to which you want to limit access. When a user performs a search query, the user can view this URL pattern in the search results if you list the user as either an allowed user or if the user is a member of an allowed group.

If more than one URL pattern matches the policy ACL, the search appliance chooses the best match in this order of precedence:

1. "Exact-Match URL Rules"
2. Forward slash "/" pattern
3. "Coarse-Grained Rules":
   - "Prefix Patterns" on page 45
   - "General URL Patterns" on page 45

Exact-Match URL Rules

If there is an exact-match URL pattern, it is the best match. An exact-match URL patterns begins with a caret (^) and ends with a dollar sign ($). The following example shows an exact-match URL pattern:

```
^http://www.example.com/mypage.html$
```

Coarse-Grained Rules

The coarse-grained rules consist of:

- "Prefix Patterns"
- "General URL Patterns" on page 45

Prefix Patterns

If there is one or more matching prefix patterns, the pattern with the longest prefix is the best match. A prefix-pattern specifies a (possibly partial) domain and a prefix of the path portion of the URL. The general format of a prefix pattern is:

```
<domain>/<prefix>
```

Examples of prefix patterns:

```
sales.example.com/products/
sales.example.com/products/mypage.html
sales.example.com/
```

General URL Patterns

If the only matching URL patterns are general patterns, the best match is undefined. The search appliance chooses one pattern for the URL pattern. A general URL pattern is any pattern other than an exact-match pattern or a prefix pattern.

Examples of general URL patterns are:

Example	Description
`contains:product`	The `product` string can appear either in the host name, such as `myproduct.com`, or at the end of a URL and doesn't have to be a full word.
`regexp:sid=[0-9A-Z]+/`	The URL has to contain a URL parameter with `sid=` followed by a value that contains either a digit or capital letter. The plus means one or more characters

Allowed Users or Groups

A policy ACL rule lists each user's or group's login ID. The user who enters a search can view the URL result if either of the following conditions is true:

- The current user's name is one of the user names listed in the rule
- The current user is a member of one of the groups listed

Otherwise, the user is denied permission to view the URL. The URL does not appear in the search results.

Adding a Policy ACL

To add a policy ACL:

1. Click **Search > Secure Search > Policy ACLs**.
2. In the **URL Patterns** field, type the pattern of the URL you want to restrict.
3. Click **Create New Policy ACLs**.
4. Under **Principal Name**, type the name of a user or group that is permitted to view the URL.

5. Click the appropriate **Principal Type** (**User** or **Group**).
6. Optionally, in the **Domain** box enter the domain name for the user or group.
7. In the **Namespace/Credential Group** box, accept the default namespace/credential group for the principal or type a different namespace/credential group.
8. If the principal name and domain are case sensitive, click the **Case Sensitive?** checkbox.
9. Click **Save**.

To navigate to the previous page, click the **Back to Policy ACL list** link.

Editing a Policy ACL

To add a policy ACL:

1. Click **Search > Secure Search > Policy ACLs**.
2. Click the **Edit** link next to the policy ACL rule you want to edit.
3. Make changes to the policy ACL.
4. Click **Save**.

Deleting a Policy ACL

To delete a policy ACL:

1. Click **Search > Secure Search > Policy ACLs**.
2. Click the **Delete** link next to the policy ACL rule you want to delete.

Importing a Configuration File

You can import a text file that contains policy ACL rules. The file you import overwrites all existing policy ACL rules.

Note: Before importing a configuration file, if you have defined policy ACL rules, click **Export Search Results** to back up your rules. The exported file is in the same format as a configuration file that you can import.

The format of each rule in the file is:

```
url_pattern allowed_user_or_group
```

Each line of the file must list only one URL pattern rule, and one or more users, denoted by the `user:` prefix or groups, denoted by the `group:` prefix, as shown in the following example:

```
example.com/docsite user:jane user:sue user:wilson group:chicagodoc
group:texasdoc
mycompany.com/engsite group:eng
mycompany.com/salessite group:sales user:yvette
```

To import a file that contains policy ACLs:

1. Under **Import a Configuration File**, click **Choose File.**
2. Select the file.
3. Click **Open**.
4. Click **Import**.

Importing and Updating Policy ACLs from an Earlier Release

If you want to use policy ACLs from search appliance releases 6.8. 6.10, 6.12, or 6.14 in release 7.0, you must import the configuration file from the earlier release and update each policy ACL to the new format.

To import and update policy ACLs:

1. Import the policy ACLs from the earlier release as described in "Importing a Configuration File."
2. For each imported policy ACL, click the **Edit** link under **Matching URL Patterns**.

   Observe that **Principal Name** and **Principal Type** are imported correctly and that default values are added for the **Domain**, **Namespace/Credential Group**, and **Case Sensitive?**.
3. Update **Domain**, **Namespace/Credential Group**, and **Case Sensitive?** as appropriate for the policy ACL.
4. Click **Save**.

Searching Policy ACLs

You can perform the following types of searches from the **Policy pattern** field on the **Search > Secure Search** > **Policy ACLs** page:

- **All Rules** or **Exact-match Rules** or **Coarse-grained Rules**

  Display rules by their type—view all rules by the filter you choose, or only those that contain text that you specify in the **Policy pattern** field. Click **Search** to list the rules, rules display in alphabetic order by the rule name. The rule filters are as follows:

  - **All Rules**—List all rules or those that contain the text you specify in the **Policy pattern** field.
  - **Exact-match Rules**—List all exact-match rules or those exact-match rules that contain the text you specify in the **Policy pattern** field.
  - **Coarse-grained Rules**—List all coarse-grained rules or those coarse-grained rules that contain the text you specify in the **Policy pattern** field.
- **Find Rules for URL**

  Provide a URL and all the rules that match the URL are displayed. This search tells you which patterns match a URL. This helps you know for a given URL, which rule applies. Enter a URL pattern in the **Policy pattern** field, choose **Find Rules for URL**, and clicking **Search**. The rules are displayed in best match order. The first rule that displays applies, and is the best match and is the rule that the search appliance applies. The first rule is the one and only rule that is applied. This best match order is useful when you have two rules that match a URL and you want to find which rule applies best to the URL.

Search results appear under **Matching URL Patterns**.

Exporting Search Results

After you search policy ACLs, you can export the search results as an XML file. To export search results, click **Export Search Results**. The exported file is in the same format as an import configuration file.

The default file name is `policy_acl.xml`.

Using Credential Groups with Policy ACLs

Policy ACLs require that the identity of a user has been verified by an authentication method. A credential group can be used to authenticate a user's identity for a policy ACL. However, although you can configure multiple credential groups for a system, the search appliance only currently supports one verified identity (see "Primary Verified Identity" on page 19) from all the credential groups for policy ACLs. Generally, the **Default** credential group (see "About the Default Credential Group" on page 21) provides the primary verified identity for use with a policy ACL.

To use a credential group with a policy ACL:

- The user in the policy ACL rule must match the identity in the **Default** credential group. For example, suppose the username in the **Default** credential group is "joe." To ensure that the search appliance can use a policy ACL with this identity, ensure that there is a policy ACL rule with the user "joe."
- Check the **Requires a Username** option (see "Require a User-Name Option" on page 21) for the **Default** credential group.
- Do not rename the **Default** credential group.

Enabling Late Binding for Policy ACLs and Per-URL ACLs

In some instances, you might not want to use early binding for allow decisions, for example, if the policy ACLs or per-URL ACLs in the index don't reflect the latest changes. For situations like this, you can enable late binding for policy ACLs and per-URL ACLs.

If you enable late binding for policy ACLs and per-URL ACLs, the search appliance accepts deny decisions only for these mechanisms. For allow and indeterminate decisions, the search appliance applies each subsequent associated mechanism in the list in order until one of them returns a decision other than indeterminate.

For information about enabling late binding for policy ACLs and per-URL ACLs, click **Admin Console Help > Search > Secure Search > Flexible Authorization**.

How to Exclude Controlled-Access Content Sources from Search

When you assign credentials that allow a search appliance to crawl and index controlled-access content, it's important to consider whether the content source includes content that you don't want anyone to see. The best way to ensure that private content is never shown in search results is to exclude all private content sources from the index. Examples of controlled-access content that should be excluded from crawl and indexing include:

- Draft working directories that contain unreviewed content.

  If the search appliance has access to all directories on a server, you can find that your index contains unfinished documents that aren't meant for review. To ensure that your site users are comfortable placing content on servers that are indexed, consider creating "no crawl" directories for their rough work, and configure the search appliance to exclude all such directories from the index.

- Highly sensitive materials that should never be discovered during search.

  Because the search appliance checks for authentication and authorization before serving results, it will never show secure results to a user who does not have authorization to view the documents. Despite this, you may have some materials that are so sensitive that they require additional care.

Excluding Controlled-Access Content from the Index

To exclude private content from the index, use one or both of these methods:

- Configure your content server to define a user policy that prohibits the search appliance account from accessing those directories.
- In the Admin Console, go to **Content Sources > Web Crawl > Start and Block URLs**. Scroll down to **Do Not Follow Patterns** and enter a pattern for each URL that corresponds to private content. Any content that matches the patterns under **Do Not Follow Patterns** is excluded from the index.

Removing Controlled-Access Content from Search Results

Despite your best efforts to set exclusion patterns and define secure access policies that prevent the indexing of private content, you may discover unanticipated content that you must remove from the index. Removing content from the search appliance index takes anywhere from 30 minutes to a few hours, depending on the size and complexity of your index. To stop serving content immediately, create an exclusion rule to remove the content from the front end while you correct the index.

To stop serving undesired content immediately:

1. Log in to the Admin Console.
2. Choose **Search > Search Features** > **Front Ends**. For each front end that you have defined:
   - In the list of Current Front Ends, click **Edit** for the front end that you want to modify.
   - On the **Remove URLs** tab, enter URL patterns to exclude the undesired controlled-access content. You can enter as many URL patterns as you need to exclude all the undesired content.
   - Click **Update**. The search appliance immediately ceases serving URLs that match these patterns.
3. Load each front end and perform a query to verify that the content is no longer being served.

To permanently remove undesired content from the index:

1. Log in to the Admin Console.
2. Choose **Content Sources > Web Crawl > Start and Block URLs**. Scroll down to **Do Not Follow Patterns** and enter URL patterns that will exclude the undesired controlled-access content. You can enter as many URL patterns as you need to exclude all the undesired content.
3. Click **Save**. The search appliance removes the undesired content when the crawler next runs.
4. To verify that the content has been removed, go to **Index > Diagnostics > Index Diagnostics** and search for the removed URLs.

Customizing the Universal Login Form

By default, the **Universal Login Form** displays sections for logging in to each credential group, a **Login** button, and the Google logo. You can deploy the **Universal Login Form** with these features. However, by using the **Search > Secure Search > Universal Login Form Customization** page in the Admin Console, you can create a **Universal Login Form** that is specific to your organization.

For example, you can make the following types of changes to page elements:

- Logo—You can use your organization's logo instead of the Google Logo or remove any logo from the page.
- Font face—You can change the font face that is used on the page.
- Header and footer—If your organization's user interface achieves a uniform look by using a standard header and footer on its pages, you can maintain this look on the **Universal Login Form**.
- Submit button text—You can change the **Login** button that appears at the bottom of the form, by changing the default text string ("Login").

Alternatively, you can also upload HTML for a completely different **Universal Login Form.**

Using the Page Layout Helper

The Page Layout Helper enables you to customize the **Universal Login Form** without directly editing any HTML. The Page Layout Helper contains the areas described in the following table.

Area	Description
Logo	Enter the location and name of the logo that you want to use. You may have to type the complete URL of the logo file. Also enter the width and height in pixels of your logo image.
Font Face	Enter the name of the font family that you want to use, for example, Arial. The font face is case insensitive. If you enter a font that is not recognized, the page uses the Times font face.
Header	Paste the header code that you want to use in the box.
Footer	Paste the footer code that you want to use in the box.
Submit button text	To replace the word "Login" on the button, type the text that you want to appear in the button.
Custom Page HTML	Insert the HTML code for a completely customized login page here. Any settings in other areas, such as Logo and Font Face are ignored if this area contains customized HTML.

To open a browser window to see how the page will look when you save your changes, click **Preview**. Changes are not saved until you click **Save**.

To customize the **Universal Login Form** by using the Page Layout Helper:

1. Click **Search > Secure Search > Universal Login Form Customization**.
2. Make changes to the appropriate area of the Page Layout Helper, as described in the preceding table.
3. Optionally, preview your changes by clicking **Preview**.
4. Click **Save**.

Creating a Fully Customized Universal Login Form

The easiest way to create a fully custom **Universal Login Form** is by:

1. Setting up a search appliance to use the **Universal Login Form**.
2. Running a secure search to display the default **Universal Login Form**.
3. Using your browser's view source feature to view and save the HTML of the default **Universal Login Form**.
4. Using the HTML of the default **Universal Login Form** as a starting point for customizing your own form.

The **Universal Login Form** HTML must contain HTML form field names that match the system's expectations. For example, for a credential group named cg1:

- The user-name field should be called "ucg1"
- The password field should be called "pwcg1"

If a credential group is already satisfied at the time the **Universal Login Form** is rendered, the **Universal Login Form** attempts to disable the login fields for the already-satisfied group(s). It does this in the following two ways:

- First, the HTML for the form fields matching the above names are dynamically changed to include the "disabled" attribute and the user-name field is populated with the credential group's verified user-name (if known).
- Second, a dynamically generated inline CSS header is generated that allows you to toggle other parts of the form.

  For example, two credential groups are defined (cg1 and cg2), and cg1 is pre-satisfied, but cg2 is not, the following CSS is generated:

  ```
  <!--
  #cg1Active {display:none; }
  #cg1Inactive {display:inline; }
  #cg2Active {display:inline; }
  #cg2Inactive {display:none; }
  -->
  ```

  This allows you to create any number of additional form elements that display or not for any particular credential group, depending on whether it's being prompted or not. For example, the default **Universal Login Form** contains the following tags:

  ```
  <tr id="cg1Active"><td>Please login to cg1:</td></tr>
   <tr id="cg1Inactive"><td><span style="color:green">Logged in to cg1</span></
  td></tr>
  ```

  This allows the prompt for the credential group to explain why the credential group fields are disabled.

To upload HTML for a customized **Universal Login Form**:

1. Click **Search > Secure Search > Universal Login Form Customization**.
2. Insert the customized HTML in the **Custom Page HTML** area of the Page Layout Helper.
3. Click **Save**.

For more information about customizing the **Universal Login Form**, click **Admin Console Help > Search > Secure Search > Universal Login Form Customization**.

Chapter 3

Use Cases with Public and Secure Serve for Multiple Authentication Mechanisms

This section provides more detailed explanation of how to set up crawl for controlled-access content and how to set up the Google Search Appliance to centralize serve-time authentication.

Use Case 1: HTTP Basic or NTLM HTTP Controlled-Access Content with Public Serve

The ABC Company wants to make its controlled-access content discoverable using intranet search. The content is stored on these internal servers:

- `events.abc.int` is a simple web server that uses HTTP Basic authentication. This server contains information about internal company events.
- `announce.abc.int` is a Microsoft IIS web server that uses Integrated Windows Authentication over NTLM HTTP. This server contains announcements for employees.
- `directory.abc.int` is another Microsoft IIS server. This server provides phone and office location information about employees. For the purpose of this example, let's suppose that content from this server is best provided by a web feed.

All these servers are located on the same domain, `abc_corp`. Although authentication is required by each of these servers, this information isn't sensitive. ABC Company wants to serve the snippet results as public content, viewable by any employee. There is no reason to require the search appliance to perform document-level authentication when serving results.

ABC Company has these people who interact with this content:

- Adam, the system administrator
- Sandra, the search appliance administrator
- Eric, an employee who needs to find content

Setting up Crawl and Index

First, the system administrator creates a user account for the search appliance, called `ABCsearch`, and sets up access policies that ensure that the `ABCsearch` user account is authorized to view all files on `events.abc.int`, and `announce.abc.int`. The feed process on `directory.abc.int` has its own account with similar permissions, called `ABCfeeder`.

Next, the search appliance administrator logs into the Admin Console and performs these actions:

1. To provide the search appliance with credentials for crawl and index, Sandra opens **Content Sources > Web Crawl > Secure Crawl > Crawler Access,** and adds rows using the account names and passwords given to her by the system administrator:

For URLs Matching Pattern, Use:	Username:	In Domain:	Password:	Confirm Password:	Make Public:
https://events.abc.int/	`ABCsearch`		******	******	X
https://announce.abc.int/	`ABCsearch`	`abc_corp`	******	******	X
https://directory.abc.int/	`ABCfeeder`	`abc_corp`	******	******	X

   Here, omitting the domain for `events.abc.int` instructs the search appliance to authenticate using HTTP Basic. For all other servers in this example, the domain entry tells the search appliance to authenticate against a Microsoft IIS Server using NTLM HTTP.

   Because Basic Authentication sends credentials as base-64 encoded clear text, the patterns for `events.abc.int` all use HTTPS, which protects user names and passwords. Although the use of HTTPS is recommended for Basic Authentication, the search appliance can also authenticate over HTTP. **Make Public** is selected for all URL patterns.

2. Under **Content Sources > Web Crawl > Start and Block URLs,** Sandra clicks **Add** under **Start URLs** and adds the URL patterns `"https://events.abc.int/"` and `"https://announce.abc.int/"`.

3. Sandra also adds the URL patterns `"https://events.abc.int/"`, `"https://announce.abc.int/"`, and `"https://directory.abc.int/"` under **Follow Patterns**.

4. Finally, she clicks **Save** to save the changes.

5. She pushes a web feed to the appliance that includes the URLs from `directory.abc.int`, using the following syntax:

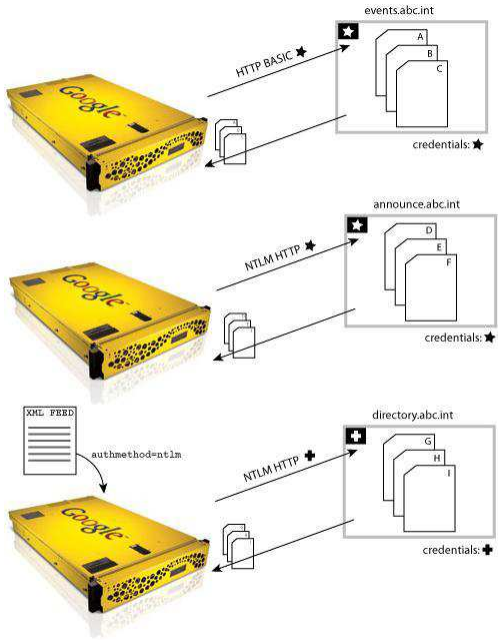
```
<record url="http://directory.abc.int/" authmethod="ntlm">
```

   Because the record has `authmethod=ntlm`, the search appliance attempts to authenticate using NTLM HTTP when crawling this content.

Now that the search appliance has access to all of ABC Company's press releases, the search appliance administrator starts the crawl and waits for the controlled-access content to appear in the index.

Populating the Index for Controlled-Access Content

During crawl, the search appliance goes through each of the content sources that have been configured, and uses the credentials under **Crawler Access** to obtain the controlled-access content.



The search appliance can use multiple protocols to crawl and index controlled-access content.

- The search appliance connects to `events.abc.int` over HTTPS. The web server asks for credentials using HTTP Basic Authentication: the search appliance provides the username "`ABCsearch`" and the password entered in the Admin Console. The web server verifies that `ABCsearch` has access to view documents on `events.abc.int`. The search appliance crawls through all documents on `events.abc.int` and adds them to the index.
- The search appliance connects to `announce.abc.int` over HTTPS. The Microsoft IIS server asks for credentials using Windows Authentication: the search appliance provides an NTLM HTTP message that contains the username "`ABCsearch`" and a response based on the password entered in the Admin Console. The IIS server verifies that `ABCsearch` has access to view documents on `announce.abc.int`. The search appliance crawls through all documents on `announce.abc.int` and adds them to the index.
- The search appliance receives a web feed that directs it to `directory.abc.int` with `authmethod=ntlm`. It connects to `directory.abc.int` over HTTPS. The Microsoft IIS server asks for credentials using Windows Authentication: the search appliance provides an NTLM HTTP message that contains the username "`ABCfeeder`" and a response based on the password entered in the Admin Console. The IIS server verifies that `ABCfeeder` has access to view documents on `directory.abc.int`. The search appliance crawls through all documents on `directory.abc.int` and adds them to the index.

Serving Controlled-Access Content to the User as Public Content

ABC Company has decided to make the search results public: the `events`, `announce`, and `directory` servers control access to their content, but employees can discover the information they need by performing a search query.

Eric is an employee of ABC Company. He wants to find an announcement about a colleague's recent promotion to Director. Eric opens the search page in a web browser and enters a query about "Maria Jones director". The search appliance performs the following steps before sending Eric to the search results page:

1. The search appliance checks to see whether any of the content sources require authorization. Although the search appliance had to provide credentials to index the content, the **Make Public?** checkbox is selected for all of ABC Company's content sources. All content in the index is labeled as public: no authorization check is required.
2. The search appliance queries the index and obtains a list of relevant results for Eric's query.
3. Eric sees search results from `events.abc.int`, `announce.abc.int`, and `directory.abc.int` that match the query "Maria Jones director". For instance, Eric finds an all-hands meeting that Maria scheduled from `events`, a notice about her promotion from `announce`, and her office phone number and location from `directory`.

When Eric clicks on one of the links in the search results page, the server that hosts the page requests a response that includes an authentication header. If Eric hasn't logged in elsewhere, he'll have to enter a username and password on a login form. Although the search appliance indexed the content as "public," the server still requires credentials before it displays the full document.

The next time that Eric clicks a link on his search results page, however, his browser forwards an authentication header based on his user name and password to the server. If all the servers in this example are on the same domain and accept the same credentials, Eric shouldn't have to log in again for as long as he keeps the browser open and the session time hasn't expired.

Use Case 2: One Set of Credentials for Multiple Authentication Mechanisms

AlphaLyon is a multi-national corporation that has various different content servers that use different authentication mechanisms.

- `http://insidealpha.com` is the URL for content protected by a single sign-on (SSO) server.
- `apacheserver.alphainside.com` is a server for content protected by a custom apache script that uses cookies from the SSO system.
- `comp.alpha.int` is a simple web server that uses HTTP Basic authentication. This server hosts some personnel information from North America.
- `pers.def.int` is a Microsoft IIS web server that uses NTLM v2 HTTP. This server hosts global personnel information, excluding North America.
- `AlphaLCM` is a connector manager with one connector instance that is used to traverse and index information (including some global personnel information) from AlphaLyon's Documentum content management system.

There is a single corporate-wide set of credentials for each employee.

Currently, when employees search for protected personnel information, they are prompted for their credentials by each authentication mechanism separately. AlphaLyon's Information Technology department has set an objective to centralize serve-time authentication for the various servers hosting personnel information. This way, users need to provide their credentials only once for content protected by several authentication mechanisms.

AlphaLyon has these people who interact with this content:

- Ashish, the search appliance administrator
- Tanya, the search appliance administrator
- Joseph, a manager who wants to view personnel information about people in his organization

This use case is based on the assumption that Tanya has added a connector for Documentum and the content from the CMS has been traversed and fed into the search appliance. For information about adding connectors, see I*ntroducing Connectors*.

Setting Up Crawl and Index

Ashish, the system administrator creates a user account for the search appliance, called `ALSearch`, and sets up access policies that ensure that the `ALSearch` user account is authorized to view all files on `comp.alpha.int`, and `pers.def.int`.

Next, Tanya sets up crawl and index of the controlled-access content by performing the following steps:

1. To provide the search appliance with credentials for crawling and indexing `comp.alpha.int`, which is protected by HTTP Basic Authentication, and `pers.def.int`, which uses NTLM HTTP, Tanya opens **Content Sources > Web Crawl > Secure Crawl > Crawler Access**.
2. Tanya adds the following rows:

For URLs Matching Pattern, Use:	Username:	In Domain:	Password:	Confirm Password:	Make Public:
`http://comp.alpha.int/`	`ALSearch`		*******	*******	
`https://pers.def.int/`	`ALSearch`	`aphalyon_corp`	*******	*******	

   Tanya uses the account name and password for `ALSearch` that was provided by Ashish, the system administrator. Note that, for `http://comp.alpha.int/`, the **In Domain** text box is cleared. This cleared checkbox instructs the search appliance to authenticate using HTTP Basic. For `http://pers.def.int/`, Tanya supplies the domain, which tells the search appliance to authenticate against the server using NTLM HTTP.

   The **Make Public** checkbox is also cleared. The search appliance has full access to the server, but labels any results from them as "secure" and requires authentication and authorization checks before displaying secure content in the search results.
3. Tanya clicks **Save**.
4. Next, Tanya needs to provide the search appliance with credentials for crawling and indexing content protected by single sign-on systems (`http://insidealpha.com` and `apacheserver.alphainside.com`), so she opens **Content Sources > Web Crawl > Secure Crawl > Forms Authentication**.
5. In the **Sample Forms Authentication protected URL** box, Tanya enters `http://insidealpha.com/inside.html`.

6. In the **URL Pattern for this rule** box, Tanya enters `http://insidealpha.com/` and clicks **Create a New Forms Authentication Rule**.

   The search appliance proxies the login form.

7. Tanya enters the credentials for the crawler user account and saves the forms authentication rule.

   The search appliance stores the rule for use in crawl for all content under `http://insidealpha.com/`. When a cookie expires, the search appliance uses the stored crawler account to request a new session cookie.

8. Next, Tanya uses the **Content Sources > Web Crawl > Secure Crawl > Forms Authentication** page to add credentials for crawling and indexing `apacheserver.alphainside.com`. In the **Sample Forms Authentication protected URL** box, Tanya enters `apacheserver.alphainside.com/alphainsider.html`.

9. In the **URL Pattern for this rule** box, Tanya enters `apacheserver.alphainside.com/` and clicks **Create**.

10. The search appliance proxies the login form.

11. Tanya enters the credentials for the crawler user account and saves the forms authentication rule.

    The search appliance stores the rule for use in crawl for all content under `apacheserver.alphainside.com/`. When a cookie expires, the search appliance uses the stored crawler account to request a new session cookie.

12. Next, to get the controlled-access content crawled and indexed, Tanya opens C**ontent Sources > Web Crawl > Start and Block URLs**.

13. Tanya clicks **Add** under **Start URLs** and adds the following URL patterns:

    - `http://comp.alpha.int/`
    - `https://pers.def.int/`
    - `http://insidealpha.com/`
    - `https://apacheserver.alphainside.com/`

14. Tanya also adds these URL patterns in the **Follow Patterns** box and clicks **Save**.

15. To check that the crawling system is currently running, Tanya opens **Content Sources > Diagnostics > Crawl Status**. The crawl status indicates that the crawl system is running.

Now that the search appliance has access to all this protected content, it can populate the index, as described in the following section.

Populating the Index with Controlled-Access Content

During crawl, the search appliance goes through each of the content sources that have been configured, and obtains the controlled-access content by using the HTTP Basic Authentication credentials configured on **Content Sources > Web Crawl > Secure Crawl > Crawler Access** and the forms authentication credentials configured **Content Sources > Web Crawl > Secure Crawl > Forms Authentication**.

For content on `comp.alpha.int`, which is protected by HTTP Basic Authentication:

1. The search appliance connects to `http://comp.alpha.int/`.

2. The web server asks for credentials using HTTP Basic Authentication.

3. The search appliance provides the username "ALSearch" and the password entered in the Admin Console.
4. The web server verifies that ALSearch has access to view documents on comp.alpha.int.
5. The search appliance crawls through all documents on comp.alpha.int and adds them to the index.

For content on pers.def.int, which is protected by NTLM HTTP:

1. The search appliance connects to pers.def.int over HTTPS.
2. The Microsoft IIS server asks for credentials using Windows Authentication.
3. The search appliance provides an NTLM HTTP message that contains the username "ALSearch" and a response based on the password entered in the Admin Console.
4. The IIS server verifies that ALSearch has access to view documents on pers.def.int. The search appliance crawls through all documents on pers.def.int and adds them to the index.

For content on http://insidealpha.com and apacheserver.alphainside.com, which are protected by forms authentication:

1. First, the search appliance connects to http://insidealpha.com/.
2. The web server asks for a session cookie.
3. the search appliance recognizes the URL pattern and provides the cookie that was set in the Admin Console under **Content Sources > Web Crawl > Secure Crawl > Forms Authentication**.
4. The web server verifies that crawler has access to view documents in the controlled access directory.
5. The search appliance crawls through all documents on http://insidealpha.com/ and adds them to the index. Because these documents were accessed through a forms authentication rule with **Make Public** cleared, they are labeled as **"**secure**"** in the index.
6. Next, the search appliance connects to apacheserver.alphainside.com/ and repeats steps 2 through 5 by interacting with the apache server.

When the crawl completes, the index contains content from the sources.

Setting Up Serve

To centralize serve-time authentication for the protected content, Tanya, the system administrator, configures the Default credential group:

1. First, to add the single sign-on server http://insidealpha.com to the credential group, Tanya opens **Search > Secure Search > Universal Login Auth Mechanisms > Cookie**.

   Because the Default credential group is already selected, Tanya does not need to select a credential group from the pull-down menu.
2. Tanya types http://insidealpha.com/inside.html, a sample URL for the site, in the **Sample URL** box. Options for adding another cookie-based domain appear on the page. The Default credential group is already selected.
3. Tanya clicks **Save.**
4. Next, to add apacheserver.alphainside.com, Tanya types apacheserver.alphainside.com/alphainsider.html, a sample URL for the content protected by a custom apache script, in the **Sample URL** box and clicks **Save**.

5. Next, to add the `comp.alpha.int` web server, which uses HTTP Basic authentication, to the credential group, Tanya clicks the **HTTP** tab.

   The Default credential group is already selected.

6. Tanya types `http://comp.alpha.int/na.html`, a sample URL for the site in the **Sample URL** box, and clicks **Save**.

   Options for adding another HTTP-based domain appear on the page. The Default credential group is already selected.

7. To add `pers.def.int`, which uses NTLM HTTP authentication, Tanya clicks the **NTLM** checkbox, types `pers.def.int/emea.html` in the **Sample URL** box and clicks **Save**.

8. Finally, to add the connector manager to the credential group, Tanya clicks the **Connectors** tab.

   The Default credential group is already selected.

9. Tanya types AlphaLCM, the name of the registered connector manager, in the **Connector Manager Name** box and clicks **Save**.

Serving Controlled-Access Content to a User with One Set of Credentials

Joseph is a manager who wants to gather all the personnel records for Pat Smith, an employee who recently joined Joseph's group from another department. Several systems in the Default credential group contain information about Pat Smith.

The following steps give an overview of the process of serving controlled-access content with Default credential group configured.

1. Joseph opens the search page in a web browser, enters a query for "Pat Smith," clicks the **public and secure content** radio button, and clicks **Search**.

2. The **Universal Login Form** checks the existing cookies that Joseph already has to see whether the credential group is already satisfied.

   The authentication mechanisms return a "rejected" response, meaning that the credential group is not satisfied.

3. The search appliance prompts Joseph by presenting the **Universal Login Form**.

4. Joseph enters his username and password on the **Universal Login Form** and clicks **Login**.

5. The search appliance applies Joseph's credentials to the systems in the Default credential group and checks each sample URL for access.

   None are rejected and the Default credential group is satisfied.

6. The search appliance queries the index and obtains a list of relevant results for Joseph's query.

7. The search appliance checks the list to see whether any of the results require authorization and filters the results based on which results Joseph is authorized to view.

8. The search appliance directs Joseph's browser to a search results page that contains all results that match the query "Pat Smith" that Joseph is authorized to view.

Use Case 3: Two Sets of Credentials for Two Connectors

AlphaLyon, from use case 2 (see "Use Case 2: One Set of Credentials for Multiple Authentication Mechanisms" on page 55), has acquired ABC company, from use case 1 (see "Use Case 1: HTTP Basic or NTLM HTTP Controlled-Access Content with Public Serve" on page 52). Content for the merged companies is managed by two different content management systems (CMSs).

- AlphaLyon's content is managed by the ECM Documentum Content Management System
- ABC company's legacy content is managed by Open Text Livelink ECM

Employees of the merged companies have two corporate-wide sets of credentials:

- 80% of the merged company's employees have credentials in AlphaLyon's system.
- 25% of the employees have credentials in ABC company's system.
- 5% of the employees have credentials in both systems.

AlphaLyon's IT department wants to centralize serve-time authentication for both systems, using both sets of credentials.

AlphaLyon has these people who interact with this content:

- Tanya, the search appliance administrator
- Leslie, a employee who joined AlphaLyon as a result of the merger who has credentials in both systems and who wants to view information from both systems

This use case assumes that Tanya has added connectors for Documentum and Livelink and the content from the CMS's has been traversed and fed into the search appliance. For information about adding connectors, see *Introducing Connectors*.

Creating a Credential Group

Tanya needs to configure two credential groups, one credential group for each of the connectors. However, because she is going to configure the Default credential group for Documentum, she only needs to create one additional credential group, for Livelink.

1. Tanya opens **Search > Secure Search > Universal Login**.
2. Tanya creates a credential group for Livelink by typing the name for the new credential group, `ABCLivelink`, in the **Credential Group Name** box.
3. Tanya types a display name for the new credential group in **Credential Group Display Name.**
4. Tanya does not click **Require a user-name for this credential group?** because no ACLs need it.
5. Tanya checks **Group is optional?** because not everyone has a login to this credential group.
6. Tanya clicks **Save**.

Adding Connectors to the Credential Groups

Next, Tanya configures the Default credential group and the ABCLivelink credential group by adding the connectors to each group:

1. First, to add the Documentum connector to the Default credential group, Tanya clicks **Search > Secure Search > Universal Login Auth Mechanisms > Connectors**.

   The Default credential group is already selected.

2. Tanya types `AlphaCM`, a mechanism name for this entry in the **Mechanism Name** box.

3. Tanya selects the connector instance to be used in the **Connector Name** box and clicks **Save**.

4. Next, to add the Livelink connector to the `ABCLivelink` credential group, Tanya creates a new entry by selecting the `ABCLivelink` credential group from the pull-down menu, typing a **Mechanism Name**, and clicking **Save**.

Serving Controlled-Access Content to a User with Two Sets of Credentials

Leslie is an employee who works on the "Island" project. She began working on this project in ABC company and continues to work on it after the merger. Both the Documentum and Livelink CMS have information about this project. Leslie wants to view information about project Island from both systems.

The following steps give an overview of the process of serving controlled-access content with two credential groups (Default and ABCLivelink) configured.

1. Leslie opens the search page in a web browser and enters a query for "Island," clicks the **public and secure content** radio button, and clicks **Search**.

2. The **Universal Login Form** checks to see whether the two credential groups are already satisfied.

   The authentication mechanisms return "rejected" responses, meaning that neither of the credential groups are satisfied.

3. The search appliance prompts Leslie for her user credentials (user name and password) for both systems by presenting the **Universal Login Form** with two logins—one for the system in the Default credential group and one for the system in the ABCLivelink credential group.

4. Leslie enters her two usernames and passwords on the **Universal Login Form** and clicks **Login**.

5. The search appliance checks her passwords with the connector managers.

   Leslie correctly entered her credentials for the system for the Default credential group but mistyped her password for the system in the ABCLivelink credential group. The Default credential group is satisfied, but the ABCLivelink credential group is not satisfied.

6. The search appliance again prompts Leslie for her credentials for the system in the ABCLivelink credential group by presenting the **Universal Login Form**.

   Because the Default credential group is already satisfied, its login is disabled (grayed-out)

7. Leslie re-enters her username and password for the system in the ABCLivelink credential group, this time correctly.

8. The search appliance checks her password with the connector manager. The credential group is satisfied.

9. The search appliance queries the index and obtains a list of relevant results for Leslie's query.
10. The search appliance checks the list to see whether any of the results require authorization and filters the results based on which results Leslie is authorized to view.
11. The search appliance directs Leslie's browser to a search results page that contains all results that match the query "Island" that Leslie is authorized to view.

Use Case 4: Windows Authentication with Kerberos Tickets for Secure Serve

AlphaLyon has decided to upgrade older servers and implement a new security policy that uses Integrated Windows Authentication (IWA) on all machines throughout their internal domain. The domain controller is a Windows server named `hal.alphalyon.com`.

AlphaLyon is going to upgrade the following servers:

- `products.alphalyon.int` is a simple web server that uses HTTP Basic authentication. This server contains information about the company's products.
- `news.alphalyon.int` is a Microsoft IIS web server that uses NTLM HTTP. This server contains news announcements.
- `emp.alphalyon.int` is another Microsoft IIS server that uses NTLM HTTP. It provides internal information about employees, such as email addresses and phone numbers.
- `sales.alphalyon.int` is a web server that uses HTTP Basic authentication. This server stores general information used by everyone on the sales team.
- `customers.alphalyon.int` is a Microsoft IIS server that uses NTLM HTTP. It stores customer directory information, such as phone numbers and addresses.

Our search appliance administrator, Tanya, wants to use Kerberos authentication to enable the search appliance to silently authenticate the user without requiring an HTTP Basic login box.

This use case is based on the following assumptions:

- Tanya has already set up crawl and index for the protected content by providing the search appliance with credentials on **Content Sources > Web Crawl > Secure Crawl > Crawler Access**.
- The following two servers have been crawled with **Make Public** selected: `products.alphalyon.int` and `news.alphalyon.int` and their content is public. Content on the other servers is secure.
- The search appliance has already indexed the protected content.

Once again, AlphaLyon has these people who interact with this content:

- Ashish, the system administrator
- Tanya, the search appliance administrator
- Eric, an employee who needs to find content
- Salim, a sales manager who needs to find information on pricing for the upcoming "AlphaLyon Product" release.

Obtaining a keytab File

Before configuring and activating Kerberos support, Tanya must obtain a Kerberos Service Key Table (keytab) file from the domain controller.

Tanya performs the following actions:

1. Tanya requests a keytab file for the search appliance from Ashish, the Windows system administrator.
2. Ashish sends Tanya a keytab file named `searchappliance.keytab`.
3. Tanya saves the keytab file on her Desktop.

Configuring and Activating Kerberos Support

Now, Tanya needs to configure the search appliance to check for a user's session ticket during serve. She also needs to activate Kerberos support:

1. Tanya opens **Search > Secure Search > Universal Login Auth Mechanisms > Kerberos**.
2. Under **Specify a Kerberos Key Distribution Center (KDC) / Windows Domain Controller (DC)**, Tanya enters `hal.alphalyon.com` in the **Kerberos KDC Hostname** box, and clicks **Save** to save the change.
3. Under **Import a Kerberos Service Key Table ("keytab") File**, Tanya clicks **Choose File** and navigates to her Desktop folder.
4. She selects the keytab file, `searchappliance.keytab`, and clicks **OK** to upload the Kerberos key table file to the search appliance.
5. She clicks **Import Kerberos Keytab File** to save the change.
6. In the section labeled **Activate IWA (Integrated Windows Authentication) / Kerberos Authentication**, she clicks **Enable Kerberos support**, and clicks **Save**. Because she is configuring Kerberos support for the Default credential group, she does not need to select a credential group from the pull-down menu.

Now that the search appliance is configured to use Kerberos authentication, any time a user requests secure content, the search appliance attempts to authenticate with the user's Kerberos session key. No additional setup is needed for secure serve.

Serving Controlled-Access Content to the User as Secure Content with Kerberos Authentication

AlphaLyon now has public and secure search results available on the search appliance, and the search appliance is able to authenticate users against a Windows Domain Controller.

Search by an Authorized User

Salim is looking for a detailed report that discusses sales figures for the new "AlphaLyon Product" release. Salim opens the search page in a web browser and enters a query for "AlphaLyon Product fall sales report".

The search appliance performs the following steps before sending Salim's browser to the search results page:

1. The search appliance queries the index and obtains a list of the most relevant results for Salim's query. The list of potential results includes announcements about the new AlphaLyon Product release (public content), as well as sales presentations and other sales collateral materials about AlphaLyon Product (secure content).

2. The search appliance filters the list of results as specified by the front end that applies to Salim's search. It applies Filters defined in **Search > Search Features** > **Front Ends** > **Filters** and excludes all URLs listed in **Search > Search Features** > **Front Ends** > **Remove URLs**.

3. The sales collateral materials come from content sources that are labeled "secure". Before it can serve results for Salim's query, the search appliance needs more information.

4. The search appliance checks to see whether Salim has provided credentials that it can use. Salim's web browser obtains or validates his Kerberos ticket from the network domain controller, which is acting as a Kerberos Key Distribution Center (KDC).

5. The search appliance sends an authorization request to Salim's web browser. Because the search appliance is configured to force the use of SSL for secure search, the request is sent over HTTPS. (This configuration is recommended, but optional.)

6. Because Salim's Kerberos ticket is valid for use by the search appliance, Salim's web browser does not display the Universal Login form. His query is silently authenticated through Kerberos.

7. Salim's Kerberos ticket is used to generate a session cookie on his computer. The browser sends Salim's cookie back to the search appliance as an authentication header sent over HTTPS.

8. Using Salim's cookie, the search appliance performs an HTTP HEAD request for each of the secure documents in the list of results. If the server returns "HTTP status 401" (not authorized) for a document, or the authorization attempt is inconclusive, the document is removed from the list of potential results. Because Salim is a member of the policy group `sales`, the search appliance should be authorized to request all of the secure sales collateral materials when passing his credentials.

9. The search appliance creates a list of search result snippets and URLs that meet all of the following criteria:

   - URLs match Salim's search query.
   - URLs are not excluded by a filter in Salim's front end.
   - URLs are not excluded by a Remove URL in Salim's front end.
   - The URL is public *or* Salim has authorization to view the URL.

10. The search appliance directs Salim's browser to the search results page that contains all public and secure documents that match the query "AlphaLyon product fall sales report". Salim should see results from `products.alphalyon.int`, `news.alphalyon.int`, `emp.alphalyon.int`, `sales.alphalyon.int`, and `customers.alphalyon.int`.

When Salim clicks on one of the links in his search results page, the browser provides his Kerberos ticket in the authentication header. The next time that Salim performs a search, the search appliance recognizes his session cookie and skips directly to the HTTP HEAD request in step 8. The session cookie set by the search appliance remains valid as long as he keeps the browser open.

The search results page doesn't tell Salim how many search results match his query or display "Goooooogle" links, since that reveals how many secure documents exist in the index.

Search by an Unauthorized User

Eric isn't a member of the sales team, but he's also interested in the new AlphaLyon Product release and wants to know when the sales figures will be posted. Eric opens the search page in a web browser and enters the same query for `AlphaLyon Product fall sales report`. The search appliance performs the following steps before sending Eric's browser to the search results page:

1. The search appliance queries the index and obtains a list of the most relevant results for Eric's query. The list of potential results includes press releases announcing the new AlphaLyon Product release, as well as sales presentations and other sales collateral materials about AlphaLyon Product.

2. The search appliance filters the list of results as specified by the front end that applies to Eric's search. It applies Filters defined in **Search > Search Features** > **Front Ends** > **Filters** and excludes all URLs listed in **Search > Search Features** > **Front Ends** > **Remove URLs**.

3. The sales collateral materials come from content sources that are labeled "secure". Before it can serve results for Eric's query, the search appliance needs more information.

4. The search appliance checks to see whether Eric has provided credentials that it can use. Eric's web browser obtains or validates his Kerberos ticket from the network domain controller, which is acting as a Kerberos Key Distribution Center (KDC).

5. The search appliance sends an authorization request to Eric's web browser. Because the search appliance is configured to force the use of SSL for secure search, the request is sent over HTTPS.

6. Because Eric's Kerberos ticket is valid for use by the search appliance, Eric's web browser does not display the Universal Login Form. His query is silently authenticated through Kerberos.

7. Eric's Kerberos ticket is used to generate an encrypted session cookie on his computer. The browser sends Eric's credentials back to the search appliance as an authentication header sent over HTTPS.

8. Using Eric's cookie, the search appliance performs an HTTP HEAD request for each of the secure documents in the list of results. If the server returns "HTTP status 401" (not authorized) for a document, or the authorization attempt is inconclusive, the document is removed from the list of potential results. Because Eric isn't a member of the policy group `sales`, the search appliance fails its authorization check using Eric's credentials. It removes all of the secure sales collateral materials from the list of potential results.

9. The search appliance creates a list of search result snippets and URLs that meet all of the following criteria:

    - URLs match Eric's search query.
    - URLs are not excluded by a filter in Eric's front end.
    - URLs are not excluded by a Remove URL in Eric's front end.
    - The URL is public *or* Eric has authorization to view the URL.

10. The search appliance directs Eric's browser to the search results page that contains all public documents that match the query "AlphaLyon product". Eric should see results from `products.alphalyon.int` and `news.alphalyon.int`, but unlike Salim, he doesn't see any results from `emp.alphalyon.int`, `sales.alphalyon.int` or `customers.alphalyon.int`.

The search results page doesn't tell Eric how many search results match his query or display "Goooooogle" links, since that reveals how many secure documents exist in the index.

Chapter 4

Cookie-Based Authentication Scenarios

This section provides detailed explanations of how selecting different options in the Admin Console affects the process of cookie-based authentication.

Overview of Scenarios

Different organizations set up cookie-based authentication rules for the Google Search Appliance's Universal Login in a variety of different ways. The selections that you, as a search appliance administrator, make by using the Admin Console depend on your system's capabilities and your organization's requirements.

For example, an organization might have a relatively simple system where, when a user does not have the correct credentials for a content server, the content server redirects the search appliance to a login system for log in, then the login system's server redirects the search appliance back to the content server after login. This instance is described in "Scenario 1: Normal Forms Authentication" on page 71.

In a different organization, the system might be more complex. For example, the system might require redirecting to one URL to get cookies and then to another URL to get a verified identity. This instance is described in "Scenario 3: Cannot Use Universal Login Form and Need Identity Verified Silently" on page 74.

Although this document does not cover every possible variation of setting up cookie-based authentication rules, it provides the following scenarios, which represent just a few possible configurations that might apply to your system's capabilities and your organization's requirements:

Each scenario contains detailed information about the interactions between the user, the search appliance, the content server (sample URL, see "Sample URL" on page 68) and login server (redirect URL, see "Sample URL Redirect to Login Form" on page 68) that take place in the different configurations. Read the scenarios so that you can decide which configuration best matches your system's capabilities and your organization's requirements.

Cookie-Based Authentication Options

To set up cookie-based authentication, you, as a search appliance administrator, use the following options on the **Search > Secure Search > Universal Login Auth Mechanisms > Cookie** page in Admin Console:

- "Sample URL" on page 68
- "Sample URL Redirect to Login Form" on page 68
- "Redirect URL" on page 68

The effects of choosing these options depend on how your system is configured, and whether your system is set up for silent authentication (see "Silent Authentication" on page 69) and cookie cracking (see "Cookie Cracking" on page 69).

Each of the scenarios in this document explains the best combination of options to choose for the situation that the scenario illustrates. The following table shows which selections and system configurations are involved in each scenario.

Sample URL	Sample URL Redirect to Login Form	Redirect URL	Silent Authentication	Cookie Cracking	Used In
✔	✔				"Scenario 1: Normal Forms Authentication" on page 71
✔		✔			"Scenario 2: Cannot Use Universal Login Form" on page 72
✔		✔	✔	✔	"Scenario 3: Cannot Use Universal Login Form and Need Identity Verified Silently" on page 74
		✔			"Scenario 4: Cannot Provide a Sample URL" on page 75
✔	✔		✔	✔	"Scenario 5: Necessary Cookie is Available for Getting a Verified Identity" on page 76
✔	✔				"Scenario 6: Use an HTTP Basic Challenge to Get Cookies" on page 78
✔		✔	✔		"Scenario 7: Use an NTLM HTTP Login Page to Get Cookies" on page 79

The following sections provide overviews of each of the options listed in the table.

Sample URL

A sample URL is any page that should not be displayed unless the user who requests the content is logged in and authorized to view it. A sample URL enables the search appliance to detect if a user is logged in, and, if so, avoid the authentication page.

An example of a sample URL is `http://it.abcreports.com/status.html`. To detect if a user is logged in, the search appliance sends an HTTP GET message to the sample URL.

If a user is logged in, the sample URL's content server, `it.abcreports.com`, returns a 200 response to the search appliance. The 200 response indicates that the request has succeeded. If the user is not logged in, the content server returns a redirect response to the search appliance (see "Sample URL Redirect to Login Form" on page 68).

Google recommends that you provide a sample URL whenever possible because it enables a quick and efficient authentication check.

To specify a sample URL, enter it in the **Sample URL** box on the **Search > Secure Search > Universal Login Auth Mechanisms > Cookie** page.

Sample URL Redirect to Login Form

If sample URL (see "Sample URL" on page 68) authentication fails, the content server can return a redirect response to the search appliance. The redirect response leads the search appliance to a single sign-on (SSO) system login form.

For example, the sample URL, `http://it.abcreports.com/status.html`, can redirect the search appliance to an SSO login form at `http://abcreports.com/login/login.html`. The search appliance can automatically log in to the form by using credentials of the credential group associated with the forms authentication mechanism.

However, for automatic login to occur, the login form must not contain any JavaScript that is critical to its submission. Otherwise, the search appliance cannot automatically log in to it.

To enable the sample URL to send a redirect response that leads to a login form, check **When sample URL fails, expect the sample page to redirect to a form, and log in to that form** on the **Search > Secure Search > Universal Login Auth Mechanisms > Cookie** page.

Redirect URL

You, as a search appliance administrator, can specify a redirect URL for the search appliance to use instead of the one supplied by the sample URL. In this case, the search appliance is redirected to URL that you specify, which can authenticate the user.

For example, suppose you specify `http://insideabcreports.com/login/login.html` as the redirect URL. When authentication at the sample URL fails, the search appliance redirects the user to the SSO login form at `http://insideabcreports.com/login/login.html`, where it can automatically log in.

If you supply a redirect URL, the authentication mechanism changes significantly. In non-redirect mode, the search appliance transfers a username / password from the **Universal Login Form** to a login form found when attempting to retrieve the sample URL. With a redirect URL, the search appliance will automatically redirect to that URL. The service at that URL can then authenticate the user in whatever way it wishes. Upon completion of that authentication, the service at the redirect URL should grant a cookie to the user which provides access to secure content (and to the sample URL, if provided), and redirect the user back to the search appliance.

If a sample URL is provided, it allows the search appliance to skip the redirect if the user already has cookies that provide access to the sample URL. A sample URL also allows verification of the user cookies upon return from the sample URL service.

Possible advantages of redirect URL authentication:

- The user's password is never sent to the search appliance.
- The redirect URL server can interact directly with the user. This can facilitate login scenarios where the user's browser must perform operations (such as evaluating complex JavaScript) that the search appliance form-filling emulator cannot perform.

Disadvantages of redirect URL authentication:

- It is generally slower than standard cookie-based forms authentication.
- It requires setting up the server for the redirect URL to respect the return URL parameter, which gives the server for the redirect URL information about the quickest path back to the search appliance.
- It does not result in a verified user-name unless the sample URL is also a cookie cracker.

On balance, Google does not recommend using a redirect URL as a preferred method of authentication.

To specify a redirect URL, enter it in the **Redirect URL** box on the **Search > Secure Search > Universal Login Auth Mechanisms > Cookie** page.

Return URL Parameter

A redirect response from the search appliance to a redirect URL includes a return URL parameter. A return URL parameter gives the server for the redirect URL information about the quickest path back to the search appliance. The server for the redirect URL follows this path when it sends a redirect response that leads back to the search appliance after it has authenticated the user.

To use a return URL parameter, the administrator of the server for the redirect URL must modify the server so that it respects a return URL parameter.

Silent Authentication

With silent authentication, users are authenticated without being directed to a login page. Inbound cookie forwarding from the content server to the search appliance can provide silent authentication without a verified identity, if the sample URL check passes.

If you require a verified identity, then silent authentication can only be achieved with cookie cracking (see "Cookie Cracking" on page 69).

Cookie Cracking

Your system might require a verified username and/or group, for example to use with authorization by means of policy ACLs, SAML, or connectors. One way of getting a verified username and/or group in addition to silent authentication is to configure the sample URL's content server for cookie cracking (see "Sample URL" on page 68).

With cookie cracking, if a sample URL check for user credentials is successful, the sample URL's content server generates the following response HTTP headers in addition to the standard headers:

```
X-Username:value
X-Groups:value1, value2
```

where *value* becomes a verified identity for the credential group that is associated with the sample URL.

The effect of the response header is that it has "cracked" open the cookie and revealed the username and/or group(s). To use cookie cracking, the administrator of the content server must modify the server so that it returns the appropriate response header.

If more than 2000 groups are used, there can be can increase in search latency and a decrease in queries per second (QPS). To avoid this issue, limit the number of groups to 2000.

There is a 3 second timeout limit for checking the sample URL. If the response time of the host is beyond this limit, the check for user credentials is not successful.

Using Quoted-Printable Encoding in Response Headers

If special characters are used in an `X-Groups` or `X-Username` HTTP response header, the header must be encoded in UTF-8 as quoted-printable. When the search appliance receives the response header, it attempts to decode the UTF-8 quoted-printable encoding.

For example, the search appliance crawls the following content, which contains special characters:

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <meta name="google:aclusers" content="spécial"/>
    <meta name="google:aclusers" content="??"/>
    <meta name="google:aclgroups" content="spécial-group"/>
  </head>
  <body>
    Some content
  </body>
</html>
```

Because the user `"spécial"` and group `"spécial-group"` include special characters, the following encoded headers should be used:

```
X-Username: sp=C3=A9cial (for spécial)
X-Groups: sp=C3=A9cial-group (for spécial-group)
```

In contrast, for the user `"??"` and the group `"spécial-group"`, the following encoded headers should be used:

```
X-Username: =E6=97=A5=E6=9C=AC (for ??)
X-Groups: sp=C3=A9cial-group (for spécial-group)
```

If there are special characters in an `X-Groups` or `X-Username` HTTP response header that are not encoded, the search appliance is not able to parse the ACL properly. To avoid this problem, Google recommends that you always encode the headers.

Scenario 1: Normal Forms Authentication

In Scenario 1, if the sample URL check fails because the user is not yet logged in, the content server redirects the search appliance to a login system for log in, then the login system's server redirects the search appliance back to the content server after login.

Set Up for Scenario 1

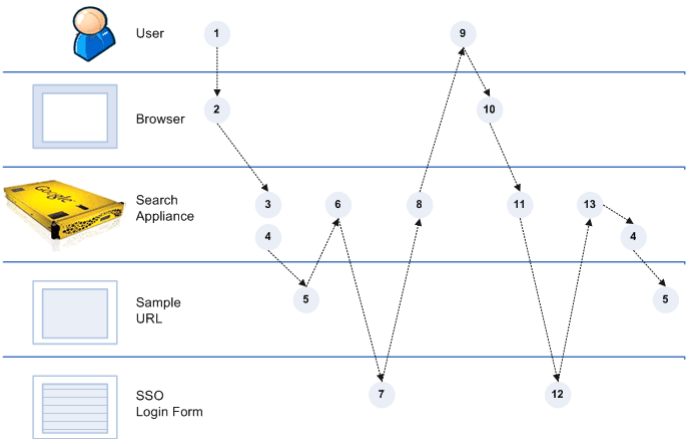
For scenario 1, set up a cookie authentication rule by performing the following tasks:

- Specify a **Sample URL**
- Check **When sample URL fails, expect the sample page to redirect to a form, and log in to that form**

The redirect to the login form is provided by the sample URL page response, so do not specify it in the **Redirect URL** box.

Process Overview of Scenario 1

The following diagram provides an overview of the cookie authentication process in scenario 1. For explanations of the numbers in the process, see the steps following the diagram.



1. The user requests a secure search.
2. The browser sends a GET message to the search appliance.
3. The search appliance checks its own session cookie to find out if authentication was previously completed.

   The search appliance sets a session cookie the first time a browser requests a secure search.
4. If the search appliance's session cookie is still valid, the authentication phase is complete.

   If the search appliance's session cookie is not valid, the search appliance checks the content server by using the sample URL to detect whether other cookies that the browser has sent are valid.

5. If the user is logged in, the content server sends a 200 response to the search appliance and authentication is complete.

   If the user is not logged in, the content server sends a 302 redirect response to the search appliance.

6. The search appliance sends a GET message to the SSO Login Form located at the URL where it was redirected.
7. The SSO Login Form sends an empty SSO Login Form to the search appliance.
8. If the search appliance has the user credentials, it completes the SSO Login Form and sends it to the SSO system. If the search appliance does not have user credentials, it sends an empty Universal Login Form to the browser.
9. The user provides a username and password for each credential group in the Universal Login Form and submits it.
10. The browser sends the completed Universal Login Form to the search appliance.
11. The search appliance adds the username and password to the SSO Login Form and sends it to the SSO system.
12. The SSO system logs in the user, sets a cookie, and sends it with a redirect response that points the search appliance to the content server.
13. The authentication phase begins again at step 4. The search appliance checks the content server by using the sample URL to detect whether the cookie is correct.

Scenario 2: Cannot Use Universal Login Form

In scenario 2, the system cannot use the Universal Login Form. For example, if a corporate SSO login system uses JavaScript, the Universal Login Form cannot log in to it. However, the user can be redirected to a form where she can log in and get cookies.

Set Up for Scenario 2

In scenario 2, if the search appliance does not receive a 200 response from the sample URL, the search appliance redirects to the SSO Login Form so that the user can log in and get cookies.

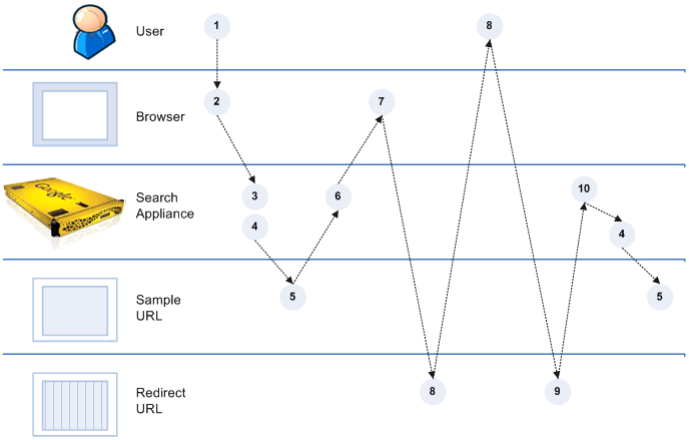
For scenario 2, set up a cookie authentication rule by performing the following tasks:

- Specify a **Sample URL**
- Specify the SSO Login Form as the **Redirect URL**

Because the sample URL does not redirect to a login form that is compatible with the search appliance, you do not need to check **When sample URL fails, expect the sample page to redirect to a form, and log in to that form**.

Process Overview of Scenario 2

The following diagram provides an overview of the cookie authentication process in scenario 2. For explanations of the numbers in the process, see the steps following the diagram.



1. The user requests a secure search.
2. The browser sends a GET message to the search appliance.
3. The search appliance checks its own session cookie to find out if authentication was previously completed.

   The search appliance sets a session cookie the first time a browser requests a secure search.
4. If the search appliance's session cookie is still valid, the authentication phase is complete.

   If the search appliance's session cookie is not valid, the search appliance checks the content server by using the sample URL to detect if other cookies that the browser has sent are valid.
5. If the sample URL check for the user credentials is successful, the content server sends a 200 response to the search appliance and authentication is complete.

   If the sample URL check is not successful, the content server sends any response except a 200 to the search appliance.
6. The search appliance sends a redirect response pointing to the redirect URL that includes a return URL parameter to the browser (see "Return URL Parameter" on page 69).

   This action forces the user to visit the Redirect URL.
7. The browser sends a GET message with a return URL parameter to the Redirect URL.
8. The user interacts with the Redirect URL and gets a cookie.
9. The Redirect URL sends a redirect response with a cookie to the address specified in the return URL parameter, which leads to the search appliance.
10. The authentication phase begins again at step 4. The search appliance checks the content server by using the sample URL to detect whether the cookie is correct.

Scenario 3: Cannot Use Universal Login Form and Need Identity Verified Silently

Scenario 3 is a variation of scenario 2 (see “Scenario 2: Cannot Use Universal Login Form” on page 72). As in scenario 2, the system cannot use the Universal Login Form. But in this scenario, you need a verified identity to use with policy ACLs. The sample URL’s server provides the verified identity.

Set Up for Scenario 3

In scenario 3, sample URL’s server is configured for cookie cracking (see “Cookie Cracking” on page 69), meaning that it can provide silent authentication and a verified username and/or groups for the credential group that is associated with the sample URL.

If the search appliance does not receive a 200 response from the sample URL, the search appliance redirects to the SSO Login Form so that the user can log in and get cookies.

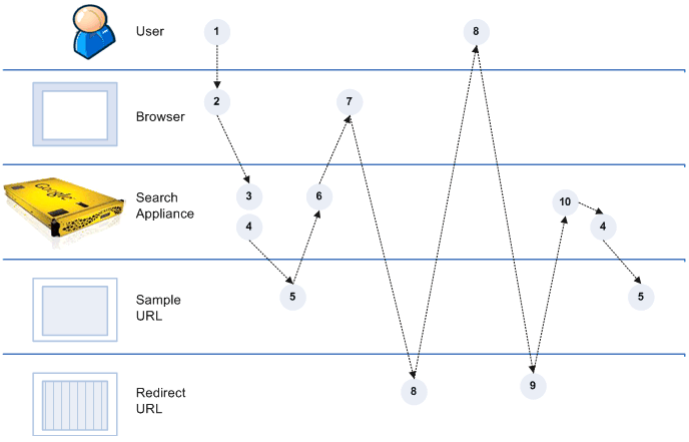
For scenario 3, set up a cookie authentication rule by performing the following tasks:

- Specify a **Sample URL**
- Specify the SSO Login Form as the **Redirect URL**

Because the sample URL does not redirect to a login form that is compatible with the search appliance, you do not need to check **When sample URL fails, expect the sample page to redirect to a form, and log in to that form**.

Process Overview of Scenario 3

The following diagram provides an overview of the cookie authentication process in scenario 3. For explanations of the numbers in the process, see the steps following the diagram.



1. The user requests a secure search.
2. The browser sends a GET message to the search appliance.

3. The search appliance checks its own session cookie to find out if authentication was previously completed.

   The search appliance sets a session cookie the first time a browser requests a secure search.

4. If the search appliance's session cookie is still valid, the authentication phase is complete.

   If the search appliance's session cookie is not valid, the search appliance checks the content server by using the sample URL to detect if other cookies that the browser has sent are valid.

5. If the sample URL check is successful, the content server generates a 200 response that includes a response HTTP header with `X-Username:value` and/or `X-Groups:value` and sends it to the search appliance.

   value becomes a verified identity for the credential group that is associated with the sample URL and authentication is complete.

   If the sample URL check is not successful, the content server sends any response except a 200 to the search appliance.

6. The search appliance sends a redirect response that includes a return URL parameter to the browser (see "Return URL Parameter" on page 69).

   This action forces the user to visit the Redirect URL.

7. The browser sends a GET message with the return URL parameter to the Redirect URL.

8. The user interacts with the Redirect URL and gets a cookie.

9. The Redirect URL sends a redirect response with a cookie to the search appliance.

10. The authentication phase begins again at step 4. The search appliance checks the content server by using the sample URL to detect whether the cookie is correct.

Scenario 4: Cannot Provide a Sample URL

In scenario 4, the system cannot provide a sample URL to enable the search appliance to detect if a user is logged in. However, the user can be redirected to a form where she can log in and get cookies.

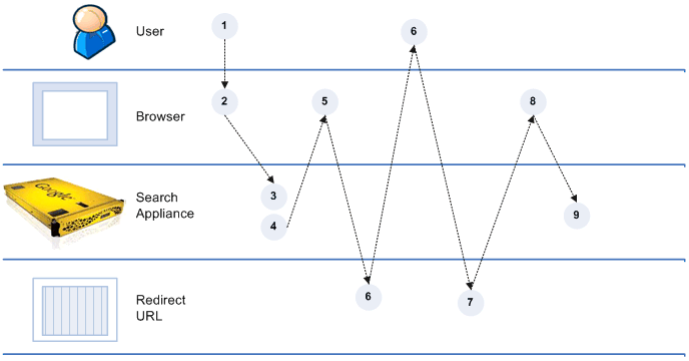
Set Up for Scenario 4

For scenario 4, set up a cookie authentication rule by specifying a **Redirect URL**.

Because your system cannot provide a sample URL, leave the **Sample URL** box blank and do not check **When sample URL fails, expect the sample page to redirect to a form, and log in to that form**.

Process Overview of Scenario 4

The following diagram provides an overview of the cookie authentication process in scenario 4. For explanations of the numbers in the process, see the steps following the diagram.



1. The user requests a secure search
2. The browser sends a GET message to the search appliance.
3. The search appliance checks its own session cookie to find out if authentication was previously completed.

   The search appliance sets a session cookie the first time a browser requests a secure search.
4. If the search appliance's session cookie is still valid, the authentication phase is complete.

   If the search appliance's session cookie is not valid, the search appliance sends a redirect response that includes a return URL parameter to the browser (see "Return URL Parameter" on page 69).

   This action forces the user to visit the Redirect URL.
5. The browser sends a GET message with the return URL parameter to the Redirect URL.
6. The user interacts with the Redirect URL and gets a cookie.
7. The Redirect URL sends a redirect response with a cookie the browser.
8. The browser redirects to the search appliance.
9. The search appliance assumes that authentication was successful and uses any cookies sent by the redirect URL in head requests.

Scenario 5: Necessary Cookie is Available for Getting a Verified Identity

In scenario 5, it is a requirement to get a verified identity for use with policy ACLs, SAML Authorization, or connectors. The system is set up so that the search appliance never forces the user to log in, but the necessary cookie is available to the search appliance. In this scenario, a portal always forces the user to log in and the search appliance gets the cookie from the portal.

Because the user is already logged in before sending a request to the search appliance, the only way to get a verified identity is by using cookie cracking (see "Cookie Cracking" on page 69).

Set Up for Scenario 5

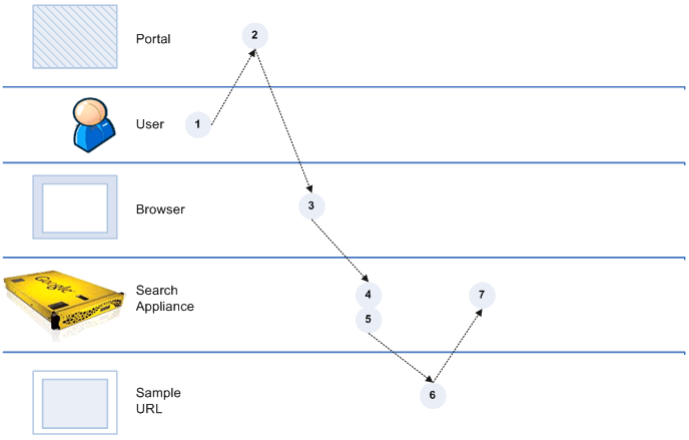
In scenario 5, the sample URL's server is configured as a cookie cracker, meaning that it can provide silent authentication and a verified identity for the credential group that is associated with the sample URL. A 200 response from the sample URL includes the `X-Username` and/or `X-Groups` HTTP response headers.

For scenario 5, set up a cookie authentication rule by specifying a **Sample URL**.

Because a cookie is provided to the browser when the user logs into the portal, you do not need to check **When sample URL fails, expect the sample page to redirect to a form, and log in to that form** or specify a **Redirect URL**.

Process Overview of Scenario 5

The following diagram provides an overview of the cookie authentication process in scenario 5. For explanations of the numbers in the process, see the steps following the diagram.



1. The user logs in to a system in the enterprise that is connected to the SSO system, such as a portal.
2. The system authenticates the user and send a cookie to the browser.
3. When the user requests a secure search, the browser sends a GET message with the cookie to the search appliance.
4. The search appliance checks its own session cookie to find out if authentication was previously completed.

   The search appliance sets a session cookie the first time a browser requests a secure search.
5. If the search appliance's session cookie is still valid, the authentication phase is complete.

   If the search appliance's session cookie is not valid, the search appliance checks the content server by using the sample URL to detect if the cookie from the portal is correct.
6. If the sample URL check is successful, the content server generates a 200 response that includes a response HTTP header with `X-Username:`*`value`* and/or `X-Groups:`*`value`* and sends it to the search appliance.
7. *value* becomes a verified identity for the credential group that is associated with the sample URL and authentication is complete.

Scenario 6: Use an HTTP Basic Challenge to Get Cookies

In Scenario 6, your system is set up to use HTTP Basic authentication. The Google Search Appliance supports both crawl-time and serve-time authentication for content protected by an HTTP Basic challenge.

To set up a search appliance for this scenario, configure crawl of the protected content and then set up a cookie authentication rule by specifying a sample URL. If the sample URL check fails because the user is not yet logged in, the content server redirects the search appliance to an HTTP Basic Login page, then the login page redirects the search appliance back to the content server after login.

Set Up for Scenario 6

For scenario 6, first configure crawl of the content protected by HTTP Basic:

1. Open the **Content Sources > Web Crawl > Start and Block URLs** page and add a URL pattern for the content protected by HTTP Basic and cookies under **Start URLs** and **Follow Patterns**.
2. Click **Save**.
3. Open the **Content Sources > Web Crawl > Secure Crawl > Forms Authentication** page and enter a sample URL for the content protected by HTTP Basic and cookies in the **Sample Forms Authentication protected URL** box.
4. Enter a URL pattern in the **URL pattern for this rule** box.
5. Click **Create**. The Admin Console displays an error page, but ignore this error page.
6. Click **Save and Close**.
7. On the **Content Sources > Web Crawl > Secure Crawl > Crawler Access** page, enter the URL pattern for the content protected by HTTP Basic and cookies under **For URLs Matching Pattern, Use**.
8. Complete the row by entering a username, and password.
9. Click **Save**.

Next, set up a cookie authentication rule on the **Search > Secure Search > Universal Login Auth Mechanisms > Cookie** page:

1. Specify a **Sample URL**.
2. Check **When sample URL fails, expect the sample page to redirect to a form, and log in to that form**.

The redirect to the HTTP Basic login page is provided by the sample URL page response, so do not specify it in the **Redirect URL** box.

Process Overview of Scenario 6

The process overview of scenario 6 is the same as the process overview of scenario 1 (see “Process Overview of Scenario 1” on page 71).

Scenario 7: Use an NTLM HTTP Login Page to Get Cookies

In scenario 7, your system is set up to use an NTLM login page for authentication. The Google Search Appliance supports both crawl-time and serve-time authentication for content protected by an NTLM login page.

To set up a search appliance for this scenario, configure crawl of the protected content and set up a cookie authentication rule by specifying a sample URL and a redirect URL. If the search appliance does not receive a 200 response from the sample URL, the search appliance redirects to the NTLM login page so that the user can log in and get cookies.

Set Up for Scenario 7

For scenario 7, first configure crawl of the content protected by NTLM HTTP and cookies:

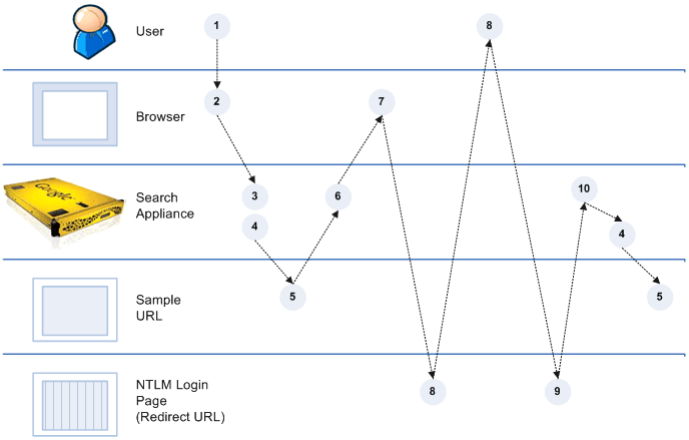
1. Open the **Content Sources > Web Crawl > Start and Block URLs** page and add a URL pattern for the content protected by NTLM HTTP and cookies under **Start URLs** and **Follow Patterns**.
2. Click **Save**.
3. Open the **Content Sources > Web Crawl > Secure Crawl > Forms Authentication** page and enter a sample URL for the content protected by NTLM HTTP and cookies in the **Sample Forms Authentication protected URL** box.
4. Enter a URL pattern in the **URL pattern for this rule** box.
5. Click **Create**. The Admin Console displays an error page, but ignore this error page.
6. Click **Save and Close**.
7. On the **Content Sources > Web Crawl > Secure Crawl > Crawler Access** page, enter the URL pattern for the content protected by NTLM HTTP and cookies under **For URLs Matching Pattern, Use**.
8. Complete the row by entering a username, domain name, and password.
9. Click **Save**.

Next, set up a cookie authentication rule on the **Search > Secure Search > Universal Login Auth Mechanisms > Cookie** page:

1. Specify a **Sample URL**.
2. Specify the NTLM login page as the **Redirect URL** You do not need to check **When sample URL fails, expect the sample page to redirect to a form, and log in to that form**.

Process Overview of Scenario 7

The following diagram provides an overview of the cookie authentication process in scenario 7. For explanations of the numbers in the process, see the steps following the diagram.



1. The user requests a secure search.
2. The browser sends a GET message to the search appliance.
3. The search appliance checks its own session cookie to find out if authentication was previously completed.

   The search appliance sets a session cookie the first time a browser requests a secure search.
4. If the search appliance's session cookie is still valid, the authentication phase is complete.

   If the search appliance's session cookie is not valid, the search appliance checks the content server by using the sample URL to detect if other cookies that the browser has sent are valid.
5. If the sample URL check for the user credentials is successful, the content server sends a 200 response to the search appliance and authentication is complete.

   If the sample URL check is not successful, the content server sends any response except a 200 to the search appliance.
6. The search appliance sends a redirect response pointing to the redirect URL that includes a return url parameter to the browser (see "Return URL Parameter" on page 69).

   This action forces the user to visit the NTLM login page (redirect URL).
7. The browser sends a GET message with the return URL parameter to the NTLM login page.
8. The user interacts with the NTLM login page and gets a cookie.
9. The NTLM login page sends a redirect response with a cookie to the address specified in the return URL parameter, which leads to the search appliance.
10. The authentication phase begins again at step 4. The search appliance checks the content server by using the sample URL to detect whether the cookie is correct.

Chapter 5

Using Trusted Applications

This chapter describes the trusted applications feature, used for secure search. It provides details about the interaction between a trusted application and the search appliance and how to configure trusted applications.

Overview of Trusted Applications

The search appliance enables trusted applications to send end-user's search requests, along with pre-validated ids when performing a secure search. The search appliance returns secure results without requiring more validation of the user.

An example of a trusted application is a web-based enterprise portal that provides secure access to search using the Google Search Appliance as its engine. With previous search appliance versions, an end user who is performing a secure search needs to supply credentials to both the portal and the search appliance. With trusted applications, the only time that end users need to supply credentials is when they log in to the portal.

Before using this feature, you enable the trusted applications feature and, optionally, register your application as a “trusted application” on the search appliance.

Trusted Applications with Registered Applications

After you register a trusted application, it can interact with the search appliance. Before sending a search request to the search appliance, the trusted application authenticates the user. After successfully verifying the user, it sends the search request, along with information about the end user's identity. Because the search appliance “trusts” the application, it returns secure results to the portal. It does not need to verify the end-user before doing so.

Trusted Applications without Registered Applications

If you enable trusted applications on the search appliance, but don’t register a trusted application, the end-user’s credentials are sent to the search appliance, which authenticates the user.

Supported Authentication Mechanisms

Take note that the search appliance only supports basic authentication and cookie authentication with trusted applications.

The search appliance does not support trusted applications with the following authentication mechanisms:

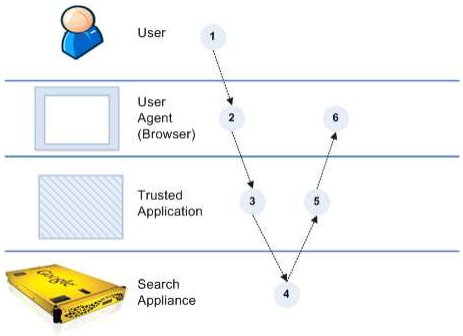
- Client certificates
- Integrated Windows Authentication/ Kerberos
- SAML
- Connectors
- LDAP

Early Binding

The trusted applications feature only supports early binding (using per-URL ACLs and policy ACLs); it does not support late binding. In early binding the security permissions are stored on the search appliance. The trusted application only needs to send the end user's ID in addition to self credentials. The search appliance only needs to authenticate the application. The end user is verified automatically.

Process Diagram

The following diagram provides an overview of the secure search API process. For explanations of the numbers in the process, see the steps following the diagram.



Before the process begins, a secure web application has been designed and configured as a trusted application on the search appliance.

1. The user signs in to the trusted application. The trusted application has the user’s verified identity.
2. The user requests a secure search from the browser.
3. The request is sent to the trusted application.

4. **If trusted applications is enabled and trusted applications are registered**: In this mode, the trusted application forwards the secure search request to the search appliance using trusted user credentials together with the end user's ID. The search appliance verifies the trusted application.

   Internally, the security manager gets the credentials from the request and verifies them. After the normal authentication check, the security manager checks to see if the request is from a trusted application by checking against the trusted application user names and/or group. If so, it creates a verified end-user session for the end user.

   There is no more credential gathering from the end user's agent. The search appliance uses the end user (`X_GSA_USER` and `X_GSA_CREDENTIAL_GROUP` headers) during the next phase (group resolution and authorization).

   If trusted applications is enabled, but no trusted applications are registered: In this mode, the user's credentials are sent to the security manager. It verifies the end user's credentials, which are used for authorization.

5. The search appliance returns secure search results to the trusted application.

6. The trusted application returns the secure search results to the user agent.

Setup for Using Trusted Applications

Set up your environment to use the trusted applications feature by performing the following tasks:

- Configuring secure search on your search appliance:
  - Setting up Universal Login by using the **Search > Secure Search > Universal Login** page, as described in "Universal Login."
  - Configuring a credential group for basic authentication or cookie authentication for trusted applications by using the **Search > Secure Search > Universal Login Auth Mechanisms** page, as described in "HTTP-Based Authentication," and "Cookie-Based Authentication."

    Take note that the trusted applications feature does not support multiple credential groups.
  - Enable early binding authorization by Adding per-URL ACLs and/or policy ACLs to the search appliance, as described in "Policy Access Control Lists."
- Enabling and registering trusted applications and adding users and/or groups, as described in the following section.
- Configuring your trusted application to send the required information to the search appliance, as described in "Configuring Your Trusted Application."

After set up is complete, start using secure search with your trusted applications.

It's important that the trusted application keep and re-use the GSA session ID. If this doesn't happen, every secure search request will create a new session. Also, the trusted application should handle the case where the GSA session ID expires.

Enabling and Registering Trusted Applications

Trusted applications is disabled by default on the search appliance.

To enable trusted applications:

1. In the Admin Console, click **Search > Secure Search > Trusted Applications**.
2. Click **Enable Trusted Applications**.

By registering a trusted application, you enable the search appliance to receive pre-validated ids from it. After a trusted application is registered, it can use the feature to interact with the search appliance. You register a trusted application by identifying it as a user or by the group where it has membership and associating it with a credential group.

If you don't register a trusted application, the end-user's credentials are sent to the search appliance, which authenticates the user.

To register trusted applications:

1. In the Admin Console, click **Search > Secure Search > Trusted Applications**.
2. Next to **Trusted Users and Groups**, click **Add**.
3. Select **User** or **Group** from the pull-down menu.
4. Select a credential group from the pull-down menu.
5. Optionally, type a domain and name for the end user.

   If the domain and name are case sensitive, click the **Case Sensitive** checkbox.
6. Click **Save**.

For more information about how to register trusted applications, click **Admin Console Help > Search > Secure Search > Trusted Applications**.

Configuring Your Trusted Application

To use the trusted applications feature, a search request from a trusted application should include the following headers:

- X_GSA_USER: Header
- X_GSA_CREDENTIAL_GROUP: Header

If no trusted applications are registered, do not use `X_GSA_USER` and `X_GSA_CREDENTIAL_GROUP` headers. Otherwise, a 502 error is returned.

X_GSA_USER: Header

The `X_GSA_USER`: header is required. It identifies the end user that the trusted application is performing the search for. This field can also include a domain, as shown in the following examples:

```
X_GSA_USER:user1
X_GSA_USER:user1@my_company.com
```

X_GSA_CREDENTIAL_GROUP: Header

The `X_GSA_CREDENTIAL_GROUP`: header is required. It identifies the credential group that the end user is associated with, as shown in the following example:

```
X_GSA_CREDENTIAL_GROUP:TAUsersCredGroup
```

Search Query Formats

The following sample queries show the format of a search query sent from a trusted application to the search appliance.

Cookie Authentication

If cookie authentication is used, the search query can be in one of the following formats:

```
curl -b "<cookie_name>=<cookie_value>" --header "X_GSA_USER:<user_name>"
--header "X_GSA_CREDENTIAL_GROUP:<credential_group_name>"
"http://www.mycompany.com/search?q=YOUR_QUERY_HERE&access=a"
```

where `<cookie_value>`=application's cookie

```
curl -b "<GSA_SESSION_ID>=<session_id>;<cookie_name>=<cookie_value>"
--header "X_GSA_USER:<user_name>"
--header "X_GSA_CREDENTIAL_GROUP:<credential_group_name>"
"http://www.mycompany.com/search?q=YOUR_QUERY_HERE&access=a"
```

where `<session_id>`=application's GSA_SESSION_ID returned as a set cookie form a previous request

Basic Authentication

If basic authentication is used, the search query can be in one of the following formats:

```
curl --user user_name:password --header "X_GSA_USER:<user_name>"
--header "X_GSA_CREDENTIAL_GROUP:<credential_group_name>"
"http://www.mycompany.com/search?q=YOUR_QUERY_HERE&access=a"

curl -b "<GSA_SESSION_ID>=<session_id>" --user user_name:password
--header "X_GSA_USER:<user_name>"
--header "X_GSA_CREDENTIAL_GROUP:<credential_group_name>"
"http://www.mycompany.com/search?q=YOUR_QUERY_HERE&access=a"
```

where `<session_id>`=application's GSA_SESSION_ID returned as a set cookie form a previous request

POST Support for Long Queries

Google recommends that you use the HTTP `POST` request with trusted applications, especially if your query strings exceed the 2KB URL length limit of `GET` requests. If you use a `GET` request, the query string is truncated if it exceeds the limit. Truncation might occur when you submit dynamic navigation queries containing a large number of metadata filters. You can avoid this limitation by submitting `POST` requests instead, which have a much larger body limit (10KB). However, using the `GET` request is also an option.

Take note that, for secure searches, `POST` requests can only be used with trusted applications. They cannot be used with secure search requests that do not use trusted applications. This type of search request only uses the `GET` command.

For more information on this topic, see "Using the POST Command" in the *Search Protocol Reference*.

Index

T

U

V

W

X