

Identifier-first login

Identifier-first login allows Snowflake to identify a user *before* presenting authentication options. In this flow, Snowflake prompts the user for their email address or username only, then displays authentication options based on the identity of the user.

An identifier-first login reduces confusion and login issues by only showing users' valid authentication options. For example, identifier-first login can do the following:

- In an environment that uses [multiple identity providers](#), it can restrict single sign-options to include only those identity providers that are associated with the user.
- When combined with an [authentication policy](#), it can hide the password option for users who need to be using an identity provider to authenticate.

For examples of how authentication policies and the identifier-first login can be combined to customize the login experience for users, see [Combining identifier-first login with authentication policies](#).

A user with the ACCOUNTADMIN role can use the ENABLE_IDENTIFIER_FIRST_LOGIN parameter to enable the identifier-first login flow for an account:

```
USE ROLE ACCOUNTADMIN;
```

```
ALTER ACCOUNT SET ENABLE_IDENTIFIER_FIRST_LOGIN = true;
```



Sign in to DF97787

Username



Continue

We process your personal information according to our [Privacy Notice](#)



Sign in to DF97787

EVERETT

Password

[Forgot your password?](#)



Sign in

[Sign in as a different user](#)

We process your personal information according to our [Privacy Notice](#)

Authentication policies

Authentication policies provide you with control over how a client or user authenticates by allowing you to specify:

- Whether users must [enroll in multi-factor authentication \(MFA\)](#).
- Which authentication methods require multi-factor authentication.
- The allowed authentication methods, such as [SAML](#), passwords, [OAuth](#), or [Key pair authentication](#).
- The [SAML2 security integrations](#) that are available to users during the login experience. For example, if there are multiple security integrations, you can specify which identity provider (IdP) can be selected and used to authenticate.

If you are using authentication policies to control which IdP a user can use to authenticate, you can further refine that control using the `ALLOWED_USER_DOMAINS` and `ALLOWED_EMAIL_PATTERNS` properties of the SAML2 security integrations associated with the IdPs. For more details, see [Using multiple identity providers for federated authentication](#).

- The clients that users can use to connect to Snowflake, such as [Snowsight](#), [drivers](#), or [SnowSQL \(CLI client\)](#). The `CLIENT_TYPES` property of an authentication policy is a best effort method to block user logins based on specific clients. It should not be used as the sole control to establish a security boundary.

You can set authentication policies on the account or users in the account. If you set an authentication policy on the account, then the authentication policy applies to all users in the account. If you set an authentication policy on both an account and a user, then the user-level authentication policy overrides the account-level authentication policy.

Note

If you already have access to the identifier-first login flow, you need to migrate your account from the unsupported `SAML_IDENTITY_PROVIDER` account parameter using the `SYSTEM$MIGRATE_SAML_IDP_REGISTRATION` function.

Use cases

The following list describes non-exhaustive use cases for authentication policies:

- You want to control whether a user, all users in an account, or specific authentication methods require MFA.
- You want to control the user login flows when there are multiple login options.
- You want to control the authentication methods, specific client types, and security integrations available for specific users or all users.
- You have customers building services on top of Snowflake using Snowflake drivers, but the customers do not want their users accessing Snowflake through Snowsight or the Classic Console.
- You want to offer multiple identity providers as authentication options for specific users.

Limitations

- The `CLIENT_TYPES` property of an authentication policy is a best effort method to block user logins based on specific clients. It should not be used as the sole control to establish a security boundary.

Considerations

- Ensure authentication methods and security integrations listed in your authentication policies do not conflict. For example, if you add a SAML2 security integration in the list of allowed security integrations, and you only allow OAuth as an allowed authentication method, then you cannot create an authentication policy.
- Use an additional non-restrictive authentication policy for administrators in case users are locked out. For an example, see [Preventing a lockout](#).

Security policy precedence

When more than one type of security policy is activated, precedence between the policies occur. For example, [network policies](#) take precedence over authentication policies, so if the IP address of a request matches an IP address in the blocked list of the network policy, then the authentication policy is not checked, and evaluation stops at the network policy.

The following list describes the order in which security policies are evaluated:

1. [Network policies](#): Allow or deny IP addresses, VPC IDs, and VPCE IDs.
2. Authentication policies - Allow or deny clients, authentication methods, and security integrations.
3. [Password policies](#) (For local authentication only): Specify password requirements such as character length, characters, password age, retries, and lockout time.
4. [Session policies](#): Require users to re-authenticate after a period of inactivity

If a policy is assigned to both the account and the user authenticating, the user-level policy is enforced.

```
--To create an authentication policy
```

```
CREATE AUTHENTICATION POLICY require_mfa_authentication_policy
  AUTHENTICATION_METHODS = ('SAML', 'PASSWORD')
  CLIENT_TYPES = ('SNOWFLAKE_UI', 'SNOWSQL', 'DRIVERS')
  MFA_AUTHENTICATION_METHODS = ('PASSWORD', 'SAML')
  MFA_ENROLLMENT = REQUIRED;
```

```
--Apply to an account
```

```
ALTER ACCOUNT SET AUTHENTICATION POLICY require_mfa_authentication_policy;
```

```
--Apply to a user
```

```
ALTER USER test_user1 SET AUTHENTICATION POLICY require_mfa_authentication_policy;
```

You want to create a separate policy to admins to prevent lock out

```
--Ideally, you want a separate policy for administrators to prevent lockout. This one SHOULD allow passwords as an authentication method
CREATE AUTHENTICATION POLICY admin_authentication_policy
  AUTHENTICATION_METHODS = ('SAML', 'PASSWORD')
  CLIENT_TYPES = ('SNOWFLAKE_UI', 'SNOWSQL', 'DRIVERS')
  SECURITY_INTEGRATIONS = ('EXAMPLE_OKTA_INTEGRATION');

--Make sure you replace <administrator_name> with your admin user
ALTER USER <administrator_name> SET AUTHENTICATION POLICY admin_authentication_policy

SHOW AUTHENTICATION POLICIES;
```

Results							
Chart							
	created_on	name	database_name	schema_name	kind	owner	comment
1	2024-10-31 13:51:23.362 -0700	ADMIN_AUTHENTICATION_POLICY	SECURITY	POLICIES	AUTHENTICATION_POLICY	ACCOUNTADMIN	
2	2024-10-31 13:50:22.868 -0700	REQUIRE_MFA_AUTHENTICATION_POLICY	SECURITY	POLICIES	AUTHENTICATION_POLICY	ACCOUNTADMIN	

About network policies

By default, Snowflake allows users to connect to the service and internal stage from any computer or device. A security administrator (or higher) can use a network policy to allow or deny access to a request based on its origin. The *allowed list* of the network policy controls which requests are allowed to access the Snowflake service or internal stage, while the *blocked list* controls which requests should be explicitly blocked.

A network policy does not directly specify the network identifiers in its allowed list or blocked list. Rather, a network policy adds *network rules* to its allowed and blocked lists. These network rules group related identifiers into logical units that are added to the allowed list and blocked list of a network policy.

Important

Network policies that existed before the introduction of network rules still work. However, all new network policies should use network rules, not the `ALLOWED_IP_LIST` and `BLOCKED_IP_LIST` parameters, to control access from IP addresses. Best practice is to avoid using both ways to restrict access in the same network policy.

Workflow

The general workflow of using network policies to control inbound network traffic is:

1. [Create network rules](#) based on their purpose and type of network identifier.
2. [Create one or more network policies](#) that include the network rules that contain the identifiers to be allowed or blocked.
3. [Activate the network policy for an account, user, or security integration](#). A network policy does not restrict network traffic until it is activated.

Network rules

Network rules are schema-level objects that group network identifiers into logical units.

Snowflake features that restrict network traffic can reference network rules rather than defining network identifiers directly in the feature. A network rule does not define whether its identifiers should be allowed or blocked. The Snowflake feature that uses the network rule specifies whether the identifiers in the rule are permitted or prohibited.

The following features use network rules to control network traffic:

- [Network policies](#) use network rules to control inbound network traffic to the Snowflake service and internal stages.
- [External network access](#) uses network rules to restrict access to external network locations from a Snowflake UDF or procedure.

Supported network identifiers

Administrators need to be able to restrict access based on the network identifier associated with the origin or destination of a request. Network rules allow administrators to allow or block the following network identifiers:

- Incoming requests**
- IPv4 addresses. Snowflake supports ranges of IP addresses using [Classless Inter-Domain Routing \(CIDR\) notation](#). For example, `192.168.1.0/24` represents all IPv4 addresses in the range of `192.168.1.0` to `192.168.1.255`.
 - VPC IDs of [AWS VPC endpoints](#). VPC IDs are not supported.
 - LinkIDs of [Azure private endpoints](#). Execute the `SYSTEM$GET_PRIVATELINK_AUTHORIZED_ENDPOINTS` function to retrieve the LinkID associated with an account.

Outgoing requests Domains, including a port range.


The valid port range is 1-65535. If you do not specify a port, it defaults to 443. If an external network location supports dynamic ports, you need to specify all possible ports.

To allow access to all ports, define the port as 0. For example, `company.com:0`.

Each network rule contains a list of one or more network identifiers of the same type. The network rule's `TYPE` property indicates the type of identifiers that are included in the rule. For example, if the `TYPE` property is `IPv4`, then the network rule's value list must contain valid IPv4 addresses or address ranges in CIDR notation.

You can create a network rule and policy by using the interface or SQL. To use the interface, go to **Admin -> Security -> Network Rules**

Create Network Rule

Creating as  ACCOUNTADMIN

Network rule name

DEMORULE

Create network rule in DATAMGT.PUBLIC ▾

Add an optional comment

Type

IPv4 ▾

Mode

Ingress ▾

Identifiers

🔍 136.33.69.3

No identifiers. Use “↵” to add a new identifier to the rule.

Cancel

Create Network Rule

For this example, we made an ingress rule for all our IP address. Afterwards go to network policies and generate a policy using the rule.

Create Network Policy

Creating as  ACCOUNTADMIN

Network policy name

Demo

Add an optional comment...

Allowed

Blocked

Select rule 

New rule 

DEMORULE



0 identifiers

IPV4

INGRESS

Cancel

Create Network Policy

We can then activate the network policy to enforce the rule

Security

[+ Network Policy](#)

Network Policies

Network Rules

Sessions

Q Filter by Name

Status **All** 2 network policies

NAME	STATUS ↑	NETWORK RULES	IPS	CREATED	
DEMO	Inactive	1 Allowed / 0 Blocked	—	11/1/2024, 4:32:52 AM	⋮
EXAMPLEPOLICY	Inactive	1 Allowed / 0 Blocked	—	10/31/2024, 5:33:29 AM	

Edit

Delete

Activate On Account

This can also be achieved using SQL:

```
--Lab 3.3
--First create a network rules

CREATE NETWORK RULE my_ip_address
  TYPE = IPV4
  VALUE_LIST = (<enter ip address>,current_ip_address)
  COMMENT = 'ip range';

--Then we create the network policy
CREATE NETWORK POLICY mypolicy1 ALLOWED_IP_LIST=(my_ip_address)
  BLOCKED_IP_LIST=({<block_list>});

DESC NETWORK POLICY mypolicy1;
```

Trust Center

Note

Snowflake reader accounts are not supported.

You can use the Trust Center to evaluate and monitor your account for security risks. The Trust Center evaluates your account against recommendations specified in [scanners](#) according to a schedule, but you can change [how frequently scanners run](#). If your account violates any of the recommendations in any of the enabled scanners, then the Trust Center provides [a list of security risks](#), and information about how to mitigate those risks.

Required privileges

A user with the [ACCOUNTADMIN](#) role must grant your role the `SNOWFLAKE.TRUST_CENTER_VIEWER` or `SNOWFLAKE.TRUST_CENTER_ADMIN` [application role](#), depending on which Trust Center tab you want to access.

See the following table for information about which application roles you need for accessing specific tabs in the Trust Center:

Trust Center tab	Required application roles
Findings	<code>SNOWFLAKE.TRUST_CENTER_VIEWER</code> or <code>SNOWFLAKE.TRUST_CENTER_ADMIN</code>
Scanner Packages	<code>SNOWFLAKE.TRUST_CENTER_ADMIN</code>

For example, to create and grant a separate role for accessing the **Findings** tab, and a separate role for accessing the **Scanner Packages** tab, you can run the following commands using the ACCOUNTADMIN role:

```
--Lab 3.4

--Create permissions for adding packages
USE ROLE ACCOUNTADMIN;

CREATE ROLE trust_center_admin_role;
GRANT APPLICATION ROLE SNOWFLAKE.TRUST_CENTER_ADMIN TO ROLE trust_center_admin_role;

CREATE ROLE trust_center_viewer_role;
GRANT APPLICATION ROLE SNOWFLAKE.TRUST_CENTER_VIEWER TO ROLE trust_center_viewer_role;

GRANT ROLE trust_center_admin_role TO USER example_admin_user;

GRANT ROLE trust_center_viewer_role TO USER example_nonadmin_user;
```

Findings

The Trust Center provides a **Findings** tab that provides the following information:

- A graph of scanner violations over time, color coded by low, medium, high, and critical severity.
- An interactive list of recommendations for each violation found. Each recommendation contains details about the violation, when the scanner was last run, and how to remediate the violation.

Findings let you identify Snowflake configurations in the account that violate the requirements of [enabled scanner packages](#). For each violation, the Trust Center provides an explanation of how to remediate the violation. After you remediate a violation, the violation still appears in the **Findings** tab until the next scheduled run of the scanner package containing the scanner that reported the violation begins, or until you [run the scanner package manually](#).

You need specific a application role to access the **Findings** tab. For more information, see [Required privileges](#).

Scanners

A scanner is a scheduled background process that checks your account for security risks based on how you configured your account. Scanners are grouped together into scanner packages. Scanners contain information about what security risks they check for in your account, and the scanner package that contains them.

Scanner packages contain a description and a list of scanners that run when you [enable the scanner package](#). After you enable a scanner package, the scanner package runs immediately, regardless of the configured schedule.

By default, scanner packages are deactivated, except for the [Security Essentials scanner package](#).

Scanner packages run according to a schedule. You cannot change the schedule of the [Security Essentials scanner package](#) scanner package, but you can change the schedule of the following scanner packages:

- [CIS Benchmarks scanner package](#)
- [Threat Intelligence scanner package](#)

You must first enable a scanner package before you can change its schedule.

You need specific application role(s) to access the **Scanner Packages** tab. For more information, see the table in [requirements](#).

The following scanner packages are available:

- [Security Essentials scanner package](#)
- [CIS Benchmarks scanner package](#)
- [Threat Intelligence scanner package](#)

Enable scanner packages

To enable a scanner package, follow the steps below:

1. [Sign in to Snowsight](#).
2. Switch to a role with the [SNOWFLAKE.TRUST_CENTER_ADMIN](#) application role granted to it.
For more information about granting these roles, see [Required privileges](#).
3. In the left navigation bar, select **Monitoring » Trust Center**.
4. Select the **Scanner Packages** tab.
5. Select a scanner package from the list.
6. Select the **Settings** tab.
7. Select **Enable**.

View available scanner packages

To view available scanner packages, follow the steps below:

1. [Sign in to Snowsight](#).
2. Switch to a role with the [SNOWFLAKE.TRUST_CENTER_ADMIN](#) application role granted to it.
For more information about granting these roles, see [Required privileges](#).
3. In the left navigation bar, select **Monitoring » Trust Center**.
4. Select the **Scanner Packages** tab.
5. Optionally, select **Provider**, **Status**, or **Search** to filter the list of scanner packages available.

To start, let's create a role to add Snowflake Trust Center Admin and apply to our user.

```
--Create permissions for adding packages
USE ROLE ACCOUNTADMIN;

CREATE ROLE trust_center_admin_role;
GRANT APPLICATION ROLE SNOWFLAKE.TRUST_CENTER_ADMIN TO ROLE trust_center_admin_role;

CREATE ROLE trust_center_viewer_role;
GRANT APPLICATION ROLE SNOWFLAKE.TRUST_CENTER_VIEWER TO ROLE trust_center_viewer_role;

GRANT ROLE trust_center_admin_role TO USER <Example_admin_user>;

GRANT ROLE trust_center_viewer_role TO USER <example_nonadmin_user>;
```

Next, we want to enable the trust center packages in the setting:

Trust Center

COMPUTE_WH

Findings 3

Scanner Packages

Provider All

Status All

3 Scanner Packages

Search

NAME	PROVIDER	NUMBER OF SCANNERS	STATUS	LAST SCAN TIME
<div><div></div><div>CIS Benchmarks</div></div>	Snowflake	38	Disabled	-
<div><div></div><div>Security Essentials</div></div>	Snowflake	4	Enabled	10/31/24, 7:49:35 AM
<div><div></div><div>Threat Intelligence</div></div>	Snowflake	1	Disabled	-

Click CIS Benchmarks to enable:

Trust Center

COMPUTE_WH

Findings 3

Scanner Packages

Scanner Packages

CIS Benchmarks

Disabled

Snowflake

The CIS Snowflake Foundations Benchmark is a set of industry-recognized best practices and security configurations intended to keep Snowflake accounts secure. All CIS Benchmarks focus on technical configuration settings used to...

Enable

Scanners

38 Scanners

Search

SCANNER	SCANNER ID
Ensure single sign-on (SSO) is configured for your account / organization	CIS_BENCHMARKS_CIS1.1
Ensure Snowflake SCIM integration is configured to automatically provision and deprovision users and groups (i.e. roles)	CIS_BENCHMARKS_CIS1.2
Ensure multi-factor authentication (MFA) is turned on for all human users with password-based authentication	CIS_BENCHMARKS_CIS1.4
Ensure minimum password length is set to 14 characters or more	CIS_BENCHMARKS_CIS1.5
Ensure that service accounts use key pair authentication	CIS_BENCHMARKS_CIS1.6
Ensure authentication key pairs are rotated every 180 days	CIS_BENCHMARKS_CIS1.7
Ensure that users who did not log in for 90 days are disabled	CIS_BENCHMARKS_CIS1.8
Ensure that the idle session timeout is set to 15 minutes or less for users with the ACCOUNTADMIN and SECURITYADMIN roles	CIS_BENCHMARKS_CIS1.9
Limit the number of users with ACCOUNTADMIN and SECURITYADMIN	CIS_BENCHMARKS_CIS1.10
Ensure that all users granted the ACCOUNTADMIN role have an email address assigned	CIS_BENCHMARKS_CIS1.11
Ensure that no users have ACCOUNTADMIN or SECURITYADMIN as the default role	CIS_BENCHMARKS_CIS1.12
Ensure that the ACCOUNTADMIN or SECURITYADMIN role is not granted to any custom role	CIS_BENCHMARKS_CIS1.13
Ensure that Snowflake roles are not owned by the ACCOUNTADMIN or SECURITYADMIN roles	CIS_BENCHMARKS_CIS1.14

Select frequency for the schedule:

Enable Scanner Package

ACCOUNTADMIN

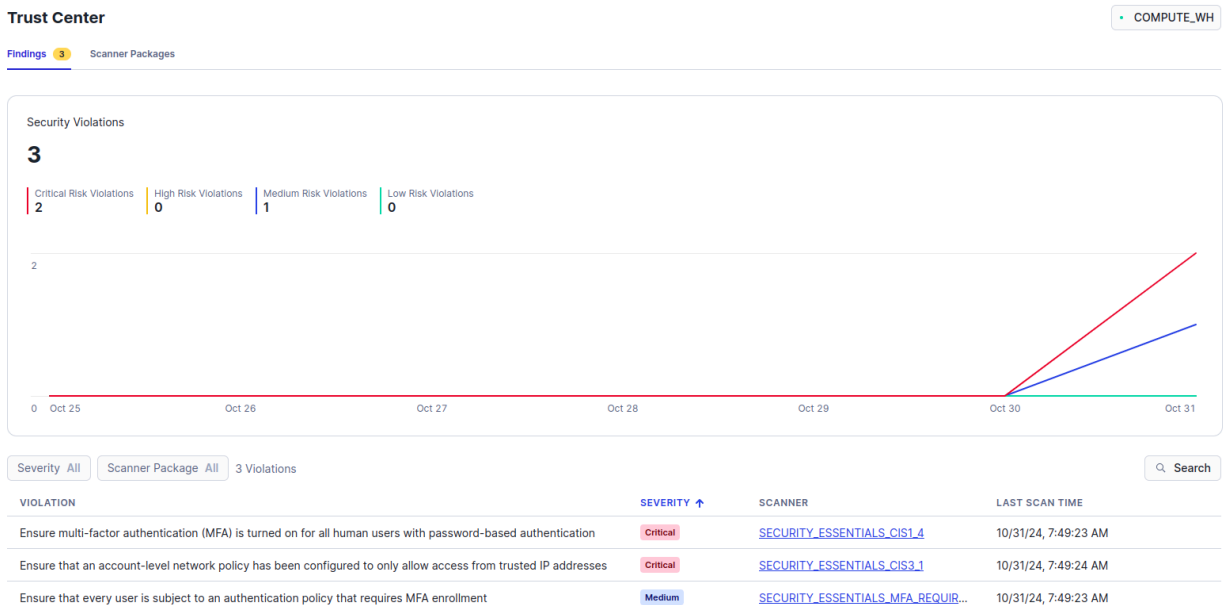
Frequency

Daily at 03:40 PM UTC

At 03:40 PM, UTC
Next run: Nov 1, 3:40 PM, UTC

Cancel Continue

Once in enabled, it will conduct scans based on the Frequency select. You can view the findings tab to see what issues need to be resolved:



Ensure multi-factor authentication (MFA) is turned on for all human users with password-based authentication →

Event ID: 102 

Summary

Remediation

Details ^

Last Scan Time 10/31/24, 7:49:23 AM

Severity

Critical

Scanner

[SECURITY_ESSENTIALS_CIS1.4](#)

Scanner Package Security Essentials

Description

Multi-factor authentication (MFA) is a security control used to add an additional layer of login security. It works by requiring the user to present two or more proofs (factors) of user identity. An MFA example would be requiring a password and a verification code

Show More ▾

Audit Result ^

3 entities were found to have violated the benchmark during the last auditing.

✎ MYUSER

✎ MYUSER2

✎ EVERETT

Run list entities sql in worksheet.

[Open a Worksheet](#) 

The remediation tab will provide recommendations, including the recommended SQL. This example recommends we require MFA.

Ensure multi-factor authentication (MFA) is turned on for all human users with password-based authentication →


Event ID: 102 

Summary Remediation

Potential Impact

If users lose access to the second factor of authentication, an account admin may need to reset their access.

Authentication policies can apply multi-factor authentication controls at the account level.



```
1 CREATE AUTHENTICATION POLICY enforce_password_mfa
2 AUTHENTICATION_METHODS = ('PASSWORD', 'SAML')
3 -- Enforce Snowflake MFA when logging in with username and
  password
4 MFA_AUTHENTICATION_METHODS = ('PASSWORD')
5 -- Require MFA enrollment when logging in with username
  and password
6 MFA_ENROLLMENT = 'REQUIRED';
7 ALTER ACCOUNT SET AUTHENTICATION POLICY
  enforce_password_mfa;
```

Show less ^

Be careful to label your programmatic users as SERVICE or LEGACY_SERVICE to avoid disruption to your operations. For specific instructions and more information check [account authentication using MFA](#). **Note:** If you use SSO authentication, you will have to check and configure MFA with your Identity Provider.