



SNOWFLAKE TECHNICAL TOOLS FOR PROTECTING SENSITIVE CUSTOMER DATA

By Seth Youssef, Security Field CTO | April 2024

Confidential. The information contained in this document is provided for informational purposes only and shall not create any representations or other obligations. Snowflake's commitments are exclusively contained within the agreement signed between your organization and Snowflake. This information is the Confidential Information of Snowflake and may not be used or disclosed except as permitted by such agreement.

CONTENTS

Document Purpose and Disclaimers	4
How To Use This Guide	5
Terminology	8
Data Privacy and Protection by Design	9
Snowflake platform security	9
Snowflake customer security and privacy controls	12
Data security capabilities	12
Data governance capabilities	12
Data privacy	13
Security of Data Processing Measures	14
A. Encryption and pseudonymization	14
Data encryption in-transit	14
Secure connectivity	14
Public connectivity	15
Private connectivity	17
Data encryption at-rest	20
Snowflake default out-of-the-box encryption	20
Snowflake Tri-Secret Secure (TSS)	22
Protecting specific columns with Snowflake column-level security	24
Split processing or multi-party processing	29
Data Pseudonymization	32
Summary of encryption and anonymization solutions	32
B. Data confidentiality, integrity, availability, and the resilience of data processing systems	33
Customer data processing confidentiality and integrity	33
High availability	34
Regional availability and resilient customer data processing	34

Cross-cloud, cross-region availability and customer data processing resilience	36
Off Snowflake backup options	38
C. The ability to restore access to sensitive data after failure	40
D. Regularity testing and evaluating data processing security	42
Regularity testing and evaluating the security of the Snowflake platform as a service	42
Regularity testing and evaluating the security of customer data	44
Account usage vs. information_schema	46
Login_History	47
Query_History	51
Object dependencies	52
Access_History	54
Sensitive columns and unauthorized access monitoring	55
Sensitive Data Ingestion Pipeline and Privacy Impact Assessment	57
Snowflake staging options	60
Summary of sensitive data ingestion pipeline	61
Sensitive Data Internal Tagging, Monitoring, and Response	63
Setting up alerts and notifications	63
Sensitive Data Incident Response (IR) Process	64
Data Subject Access Right (DSAR)	69
Right to Erasure (Forgotten, Deletion)	70
Technically define what it means to be deleted or forgotten	70
Define the deletion/forgotten time frame	71
Right to be forgotten guidance (RTBF)	71
Restricting sensitive data processing	73
Data Rectifications	74
Notification of Rectification or Erasure of Sensitive Data or Restriction of processing	75
Data Portability	76

Document Purpose and Disclaimers

Snowflake customers frequently have questions on how best to safeguard personal and other sensitive data within the Snowflake Data Cloud. This document serves as a comprehensive guide, detailing the array of features and tools available to Snowflake users. It aims to empower customers to implement robust data protection strategies for the data they upload (“**Customer Data**”), and their own customers' data if they build data products on the top of Snowflake accounts.

An additional challenge customers face is their data may fall under the purview of various jurisdiction-specific regulations, such as GDPR or CPRA. This guide seeks to bridge the gap between complex legal requirements and practical technical solutions by matching data protection mandates with detailed explanations of the specific technical controls that Snowflake customers can use to protect sensitive data objects.

Important:	While this guide is written with the GDPR in mind, it does NOT provide a legal interpretation of GDPR, nor is it intended to provide a complete GDPR solution for customers. Processes described in this guide may not satisfy a particular customer's specific regulatory and policy requirements. Each Customer must determine which technical controls are necessary to meet their obligations under applicable law based on the nature and sensitivity level of their data.
Notes:	<ul style="list-style-type: none">• This document covers Snowflake data cloud capabilities that are in general availability as of the update date on the cover. The document does not cover Snowpark Container Services and Unistore.• This document is up-to-date as of April 2024. The Snowflake product documentation provides current summaries of the Snowflake Service.

How To Use This Guide

This section matches data protection requirements with the relevant technical controls that Snowflake customers can use to protect sensitive data objects. GDPR requirements are provided as examples, but similar requirements may apply to other data protection laws.

Privacy Principle	GDPR Equivalent	Relevant Snowflake Controls
Data protection by design and by default	Data protection by design and by default GDPR Article 25	Snowflake data privacy and protection by design Sensitive data ingestion pipeline and privacy impact assessment
Security of processing	Security of processing GDPR Article 32	Security of data processing measures Sensitive data ingestion pipeline and privacy impact assessment
Pseudonymisation and encryption of personal data	Security of processing GDPR Article 32(a)	Encryption and anonymization
The ability to ensure the ongoing confidentiality, integrity, availability, and resilience of processing systems and services	Security of processing GDPR Article 32(b)	Data confidentiality, integrity, availability, and resilience of data processing systems
The ability to restore the availability and access to personal data in a timely manner in the event of a physical or technical incident	Security of processing GDPR Article 32(c)	The ability to restore access to sensitive data after failure

A process for regularly testing, assessing, and evaluating the effectiveness of technical and organizational measures for ensuring the security of the processing	Security of processing (GDPR Article 32(d))	Regularity testing and evaluating the security of the data processing
Notification of a personal data breach	Notification of a personal data breach to the supervisory authority GDPR Article 33	Sensitive data incident notification and response Sensitive data incident response (IR) process
Data protection impact assessment	Data protection impact assessment GDPR Article 35	Sensitive data ingestion pipeline and privacy impact assessment Regularity testing and evaluating the security of the data processing Sensitive data incident notification and response Sensitive data Incident Response (IR) process
Transfers subject to appropriate safeguards	Transfers subject to appropriate safeguards GDPR Article 46	Security of data processing measures Snowflake data privacy and protection by design Data encryption in-transit Data encryption at-rest Sensitive data ingestion pipeline and privacy impact assessment
Right of access by the data subject	Right of access by the data subject GDPR Article 15	Data Subject Access Right (DSAR)
Right to rectification	Right to rectification GDPR Article 16	Data rectifications

Right to be forgotten (aka Right to erasure)	Right to erasure ('right to be forgotten') GDPR Article 17	Right to erasure (Forgotten, Deletion)
Right to restriction of processing	Right to restriction of processing GDPR Article 18	Restriction of sensitive data processing
Notification obligation regarding rectification or erasure of personal data or restriction of processing	Notification obligation regarding rectification or erasure of personal data or restriction of processing GDPR Article 19	Notification of rectification or erasure of sensitive data or restriction of processing
Right to data portability	Right to data portability GDPR Article 20	Snowflake data portability
Processing of special categories of personal data	Processing of special categories of personal data GDPR Article 9	Security of data processing measures Snowflake data privacy and protection by design Data encryption in-transit Data encryption at-rest Sensitive data ingestion pipeline and privacy impact assessment
Processing of personal data relating to criminal convictions and offenses	Processing of personal data relating to criminal convictions and offences GDPR Article 10	Security of data processing measures Snowflake data privacy and protection by design Data encryption in-transit Data encryption at-rest Sensitive data ingestion pipeline and privacy impact assessment

Terminology

The following table lists terms relevant to data protection laws and their Snowflake equivalents.

Term	Definition	Snowflake equivalent
Personal Data	Any information that is related to an identified or identifiable natural person.	Snowflake processes personal data that you upload to our Data Cloud in the course of provisioning services to you. We refer to this data as <i>Customer Data</i> . Please keep in mind that Customer Data can include just about any type of data, including non-personal data. Snowflake is generally not aware of whether data is personal data or otherwise.
Data Controllers	A legal or natural person, an agency, a public authority, or any other body who, alone or when joined with others, determines the purposes of any personal data and the means of processing it.	Snowflake customers.
Data Processors	A legal or a natural person, agency, public authority, or any other body, who processes personal data on behalf of a data controller.	Such as Snowflake
Data Subjects	An identified or identifiable natural person. The natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier, such as a name, an identification number, location data, an online identifier, or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person.	The person to whom Customer data relates. This could be anyone, as it really depends on the customer. Snowflake would not have relationships with the data subjects and would generally not be aware if Customer Data related to one person versus another.

Data Privacy and Protection by Design

Note: This section discusses security controls that are potentially relevant to [GDPR article 25, Data protection by design and by default](#).

Snowflake provides rich data security, data governance, and data privacy capabilities that customers can use to apply the appropriate level of protection required by their policies.

Snowflake platform security

Snowflake is a self-managed service that has a shared responsibility model. The Snowflake Data Cloud:

- Is designed with the security of the platform from the backend in mind, and
- Provides customers with security and data protection controls that customers can use to protect their data in the platform

This topic is discussed in the following sections: [Snowflake Customer Security and Privacy Controls](#) and [Security of Data Processing Measures](#).

From the backend, Snowflake uses and applies [industry security best practices across three main pillars](#):

- [Customer Data Security and Protection Controls](#)
- [Government and Industry Data Security Compliance](#)
- Infrastructure Resiliency

Snowflake is a multi-tenant environment. This means that multiple Snowflake accounts (or Snowflake *instances*) exist in the same Snowflake Data Cloud deployment in [supported regions](#).

Currently, Snowflake implements horizontal and vertical protection and isolation between tenants. The current configuration is as shown in figure 1a.

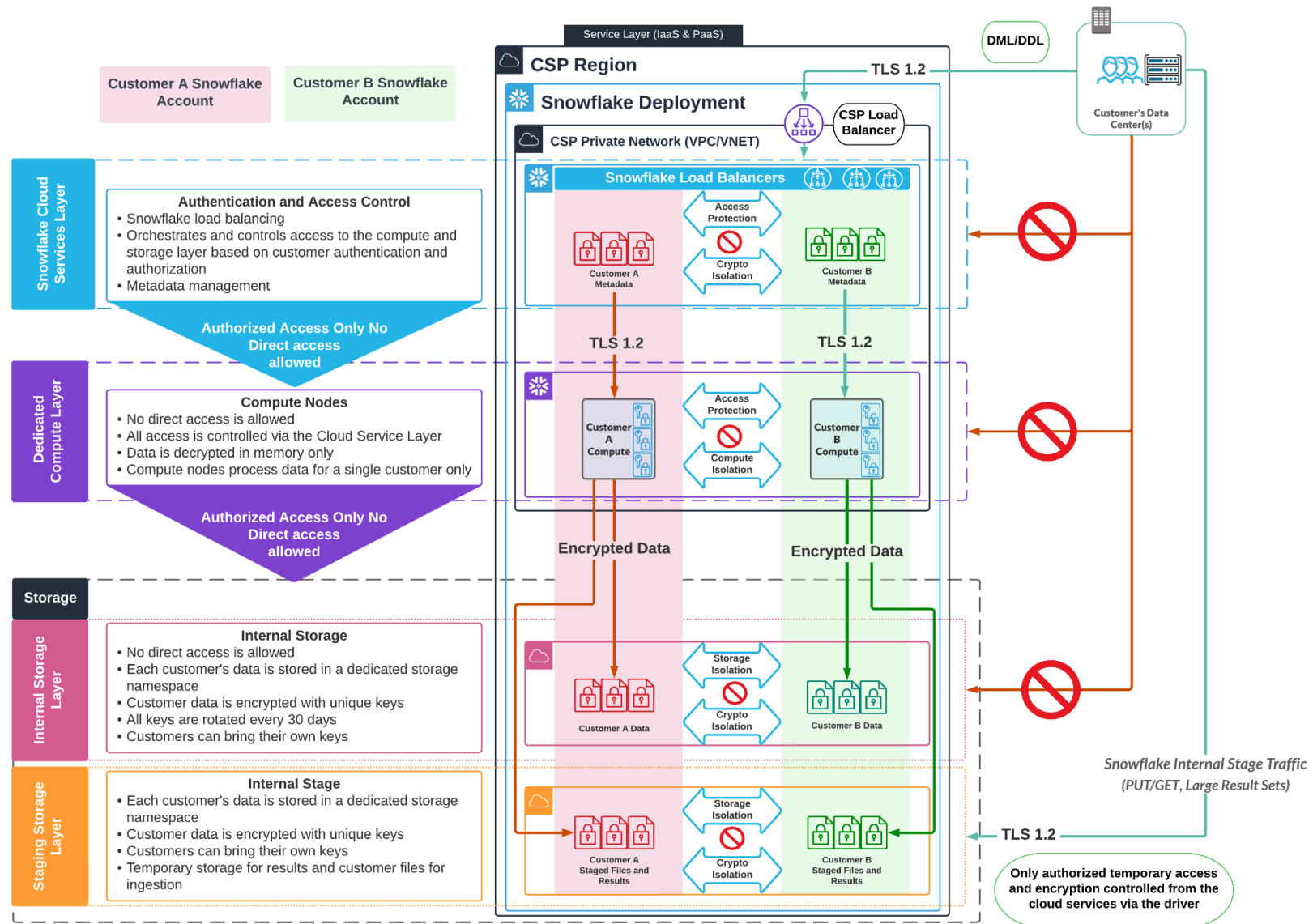


Figure 1a. Snowflake vertical and horizontal isolation

Snowflake accounts are isolated and protected:

- **Horizontally:** The [Snowflake architecture](#) separates storage, compute, and cloud services.
 - **Storage:** Snowflake provides two storage categories:
 - **Internal Storage** – Stores customer data in encrypted format. All access to this storage is controlled, authenticated, and authorized by the cloud service layer. No direct access to the internal storage is allowed from outside of the Snowflake deployment.
 - **Internal Stage** – Temporarily stores the customer's large result sets and files (via PUT/GET operations). Customer data in this storage location is encrypted. Access to this storage is controlled, authenticated, and authorized by the cloud service layer, which happens directly using customer-installed Snowflake drivers.
 - **Compute:** [Virtual warehouses](#) are customer-dedicated compute nodes while running customer queries. Data decryption and encryption happens only in the memory of the compute nodes. All access, provisioning, deprovisioning, and orchestration is authenticated and authorized via the cloud service layer. No direct access to the compute nodes is allowed from outside of the Snowflake deployment.
 - **Cloud Services Layer:** This is the entry point to the Snowflake service. It is where the cloud services manage the customer's [metadata](#), authentication, access control, high availability, load balancing, governance, optimization, and more. The customer's metadata is also encrypted.
 - **Intra-layer communications:** Data communications between cloud services, compute, and storage is always encrypted.
- **Vertically:** This is how the [Snowflake accounts](#) are isolated and protected by default. Snowflake implements multi-level protection and isolation.

Note: For more information about Snowflake security from the backend, please see the latest copy of the Snowflake [SOC2 Type 2 and other available certification reports](#).

Note: With the exception of Virtual Private Snowflake, [Snowflake editions](#) (Standard, Enterprise, and Business Critical) are multi-tenant, which means that a single deployment can host multiple Snowflake accounts belonging to different customers.

Snowflake customer security and privacy controls

Data security capabilities

Snowflake is built with security from the ground up. The Snowflake security team maintains the security of the Snowflake Data Cloud™ from the backend, while customers are provided with a wide [range of security controls](#) to further protect and control access to their data. In addition, Snowflake has earned [industry certifications](#) to give Snowflake customers confidence in the platform as stated in the [Snowflake Platform Security](#) section. Figure 1b shows the current Snowflake defense-in-depth security controls. Your data is protected by all the policies and controls configured at each layer.

Note: Please visit [Security Overview and Best Practices](#) and [Snowflake Security and Identity & Access Management for Data Applications](#) for more information about defense in depth in Snowflake.



Achieve Compliance benchmarks and Privacy goals

Figure 1b. The Snowflake security defense-in-depth security model

Data governance capabilities

Before serving the data to data consumers as ready-to-query data products, customers must put protection policies in place as determined by the customer based on the data they upload to Snowflake. Snowflake data cloud provides a growing set of [data governance capabilities](#) to help customers understand their data and then apply protection policies

that help them towards satisfying their security, compliance, and regulatory requirements (e.g. column-level encryption, tokenization, and masking). Data governance features in Snowflake provide [visibility](#) and security, and enforce data governance policies across the data pipeline without the need to move or copy data outside of Snowflake, while also providing auditing, monitoring, and lineage for data in Snowflake.

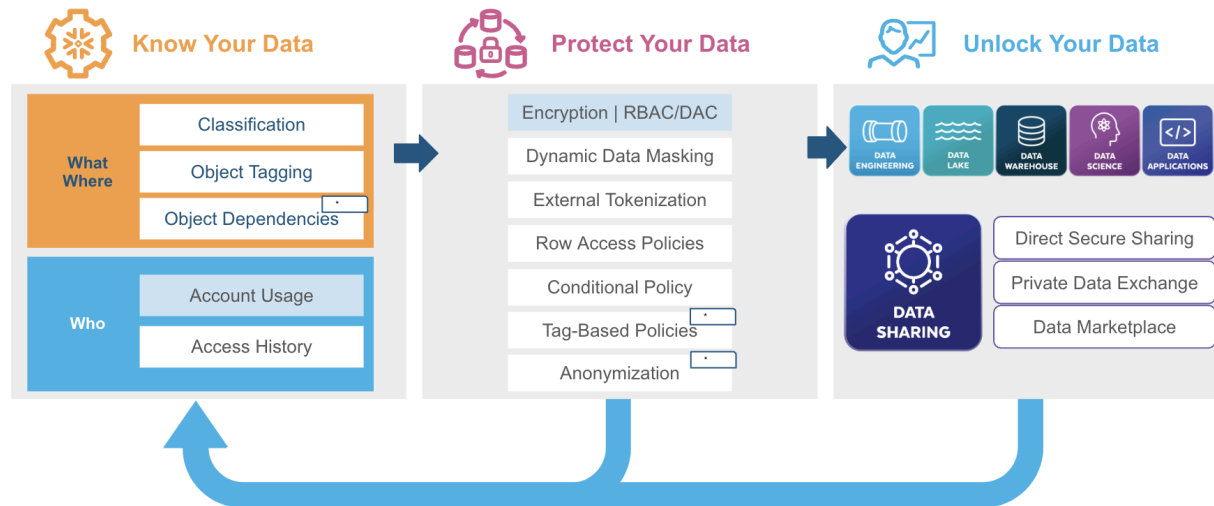


Figure 2. Snowflake data governance capabilities

Note: Capabilities marked with an asterisk are in preview at the time this paper was written.

Data privacy

Combining Snowflake platform [security](#), customer [security controls](#), data governance [policies](#), and [auditing and monitoring](#), gives customers the ability to (1) identify PII and sensitive information, and (2) apply tags and policies (such as [row access policies](#), [hashing](#), [encryption](#), [tokenization](#), [masking](#), or anonymization) to the application data tier to help achieve their organization's privacy and compliance goals.

Security of Data Processing Measures

The following sections describe how customers can protect sensitive data and help ensure the security of data processing.

- A. [Encryption and pseudonymization](#)
- B. [Data confidentiality, integrity, availability, and the resilience of data processing systems](#)
- C. [The ability to restore access to sensitive data after a failure](#)
- D. [Processes for regularity testing and evaluating data processing security](#)

A. Encryption and pseudonymization

Data encryption in-transit

Customer Data in transit to/from the Snowflake Data Cloud is always encrypted using TLS 1.2 by default, as further specified in the [Snowflake security addendum](#).

Secure connectivity

All communications within the Snowflake Data Cloud follow the same basic patterns and channels of communication as described below.

When an application connects to Snowflake via drivers, connectors, APIs, and the Snowflake web UI, it must connect to the following service endpoints to be able to use Snowflake:

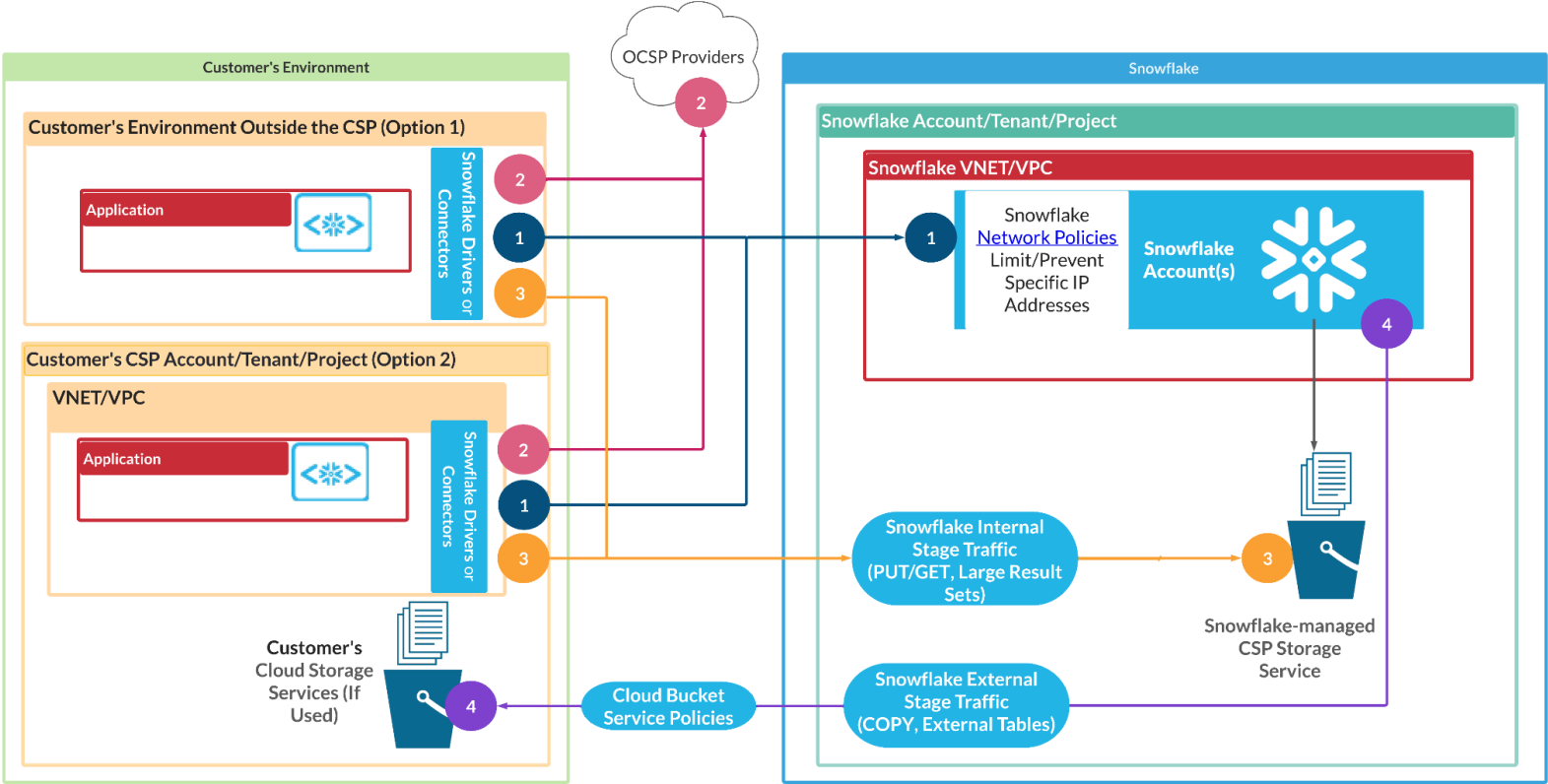
1. The **Snowflake service endpoint**, to specify the account URLs that could be [public](#) or [private](#).
2. The **Online Certificate Status Protocol (OCSP) cache endpoint**, to validate the TLS certificate. This is optional but Snowflake recommends validating the certificate via OCSP.
3. The **Internal Stage (Snowflake-managed storage)**, which is also used to return the [large results](#), as well as for PUT/GET operations on the [Internal Stage](#). This connectivity can be public or private.

Snowflake communications can flow over two main connectivity patterns: *public* and *private*. Both are discussed in the next sections.

Public connectivity

Figures 3 and 4 show the communication with the Snowflake public URL: over the internet, or over the CSP's backbone without traversing the internet.

- **Option 1, over the internet:** If the customer environment is located on a CSP platform other than the platform their Snowflake account(s) are using, communication between the customer environment and Snowflake will be over the public internet.
- **Option 2, over the CSP's backbone:** If the customer environment is located on the same CSP platform as their Snowflake account(s), the communication between the customer environment and Snowflake will be over the CSP's backbone, as stated by [GCP](#), [AWS](#), and [Azure](#). This is the case even where a Snowflake public URL and public IP address are used.



URL(s) specific to your account are obtained using SYSTEM\$ALLOWLIST

- 1 Account URL (for example, accountname.snowflakecomputing.com:443)
- 2 OCSP URL (for example, ocsp.digicert.com:80)
- 3 Internal stage (for example, Azure, GCP, or AWS bucket service URL over TLS)
- 4 Bulk data copy from customer's blob storage external stage

Figure 3. Public connectivity to Snowflake

Customers leverage Snowflake [drivers and connectors](#), and uses the communication channels shown in Table 1 below.

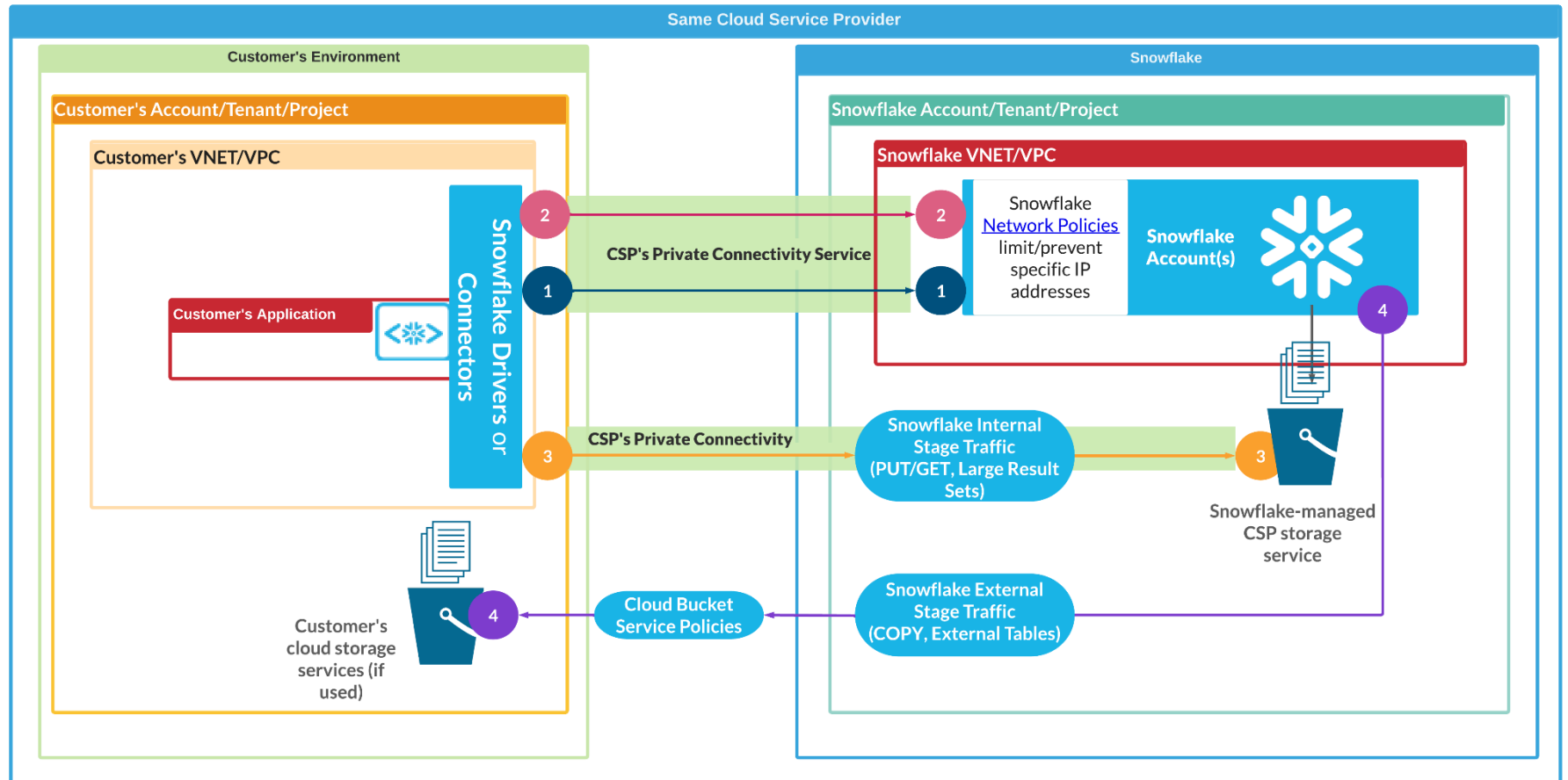
Table 1. Snowflake public communication channels and their uses

Channel	Usage	Required
Channel 1	To access Snowflake services.	Yes
Channel 2	To make a certificate-revocation check via an OCSP URL.	No, but recommended
Channel 3	To GET or PUT files and/or download query large results sets via Snowflake-managed CSP storage services.	Yes
Channel 4	<i>Optional.</i> To integrate the customer's Snowflake account with the customer's CSP storage service bucket(s). Storage bucket(s) could be provided by the same CSP that provides their Snowflake account(s), or by another CSP (GCP, AWS, Azure).	No

Private connectivity

[Business Critical Edition](#) offers private connectivity access by integrating with the underlying CSP's private connectivity offering (either [AWS PrivateLink](#), [Azure Private Link](#), or [GCP Private Service Connect](#)). Application providers can support this enhanced security option by creating a network topology to support it.

Figure 4 shows how a customer can communicate with Snowflake's private URL. Both Snowflake and the customer VPC/VNet must be on the same CSP platform because AWS PrivateLink, Azure Private Link, and GCP Private Service Connect do not support cross-cloud communications.



URL(s) specific to your account are obtained using `SYSTEM$ALLOWLIST_PRIVATELINK`

- 1 Account URL (for example, `accountname.Privatelink.snowflakecomputing.com:443`)
- 2 OCSP URL (for example, `ocsp.Privatelink.snowflakecomputing.com:80`)
- 3 Internal stage (for example, Azure, GCP, or AWS bucket service URL over TLS)
- 4 Bulk data copy from customer's blob storage external stage

Figure 4. Private Connectivity to Snowflake

Customers leverage Snowflake [driver/connectors](#), and use the communication channels shown in Table 2.

Table 2. Snowflake private communication channels

Channel	Usage	Required
Channel 1	Used to access Snowflake services over a private URL. Must be in <code>accountname.privatelink.snowflakecomputing.com</code> format. <ul style="list-style-type: none">• Azure Private Link• AWS PrivateLink• GCP Private Service Connect	Yes
Channel 2	Used to make a certificate-revocation check via an OCSP private URL.	No, but recommended
Channel 3	Used to GET or PUT files and/or download query large results sets via Snowflake-managed CSP storage services. Customers can choose to enable private connectivity to the Internal Stage .	Yes
Channel 4	Used if the customer chooses to integrate their Snowflake account with their CSP storage service bucket(s). This bucket could be provided by the same CSP that provides their Snowflake account(s), or by another CSP (either GCP, AWS, or Azure).	No

Data encryption at-rest

Snowflake default out-of-the-box encryption

Snowflake currently encrypts customer data at-rest using industry standard AES-256 encryption. This section addresses the ownership and management of encryption keys on the Snowflake service.

Figure 5 illustrates how Snowflake manages storage of Customer data. In this example, a micro-partition (MP) stores between 50 and 500 MB of uncompressed data, but because data is stored compressed, the size in Snowflake is smaller. Snowflake encrypts each micro-partition using a unique AES-256 file key that is derived at run-time in the memory of the compute nodes (aka virtual warehouses) from micro-partition metadata, and from the master key that is one level up and next in the key hierarchy.

Note: The micro-partitions are write-once, read-many files. When the customer updates a record in a table, Snowflake creates a new micro-partition and encrypts it using a new file key. The old micro-partition is kept for a period of time equaling the [Time Travel](#) setting (configurable from 0–90 days) plus the [Fail-safe](#) period (seven days non-configurable). Retention settings are covered in the [Right to Be Forgotten](#) section.

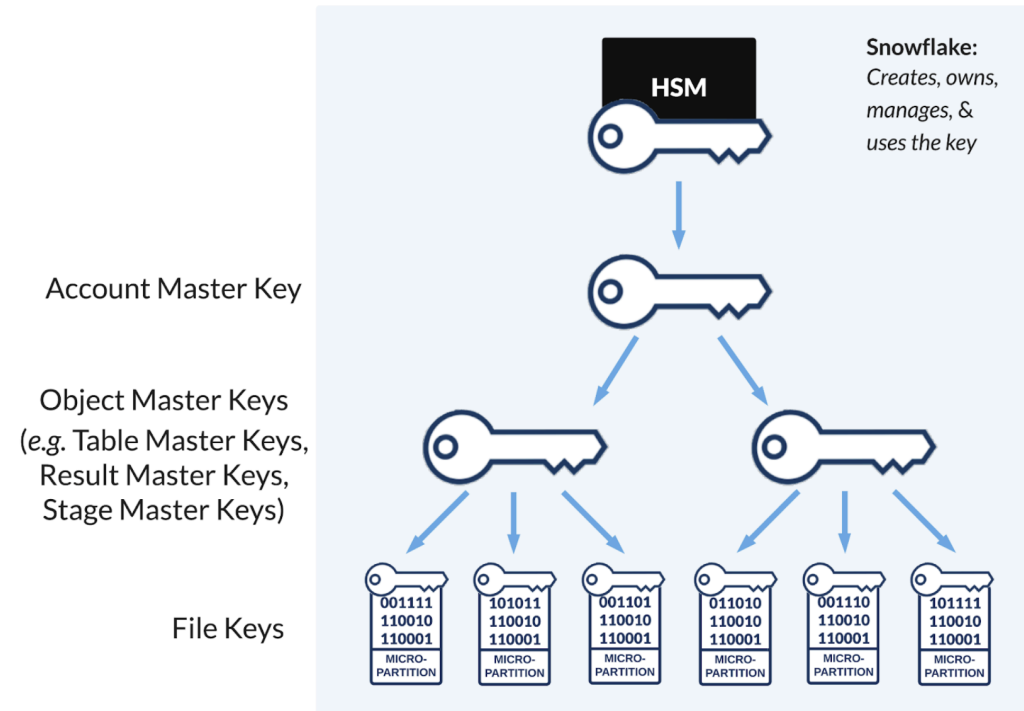


Figure 5. Snowflake Default Encryption

The next set of keys in the hierarchy are called the object master keys (OMKs). Examples of OMKs are table master keys (TMKs), stage master keys (SMKs), and results master keys (RMKs). Every table, stage, and new result has a dedicated master key. For example, if a customer has 100 tables, there will be at least 100 table master keys.

When they are not in use, object master keys are wrapped/encrypted using the account master key (AMK). Each Snowflake account/instance that the customer has, has a separate, unique, account master key (AMK). When the AMK is not in use, it is encrypted by the Snowflake root master key, which does not leave the Snowflake Cloud HSM. All encryption key wrapping/unwrapping happens only in memory.

Currently, encryption key management in Snowflake aligns with NIST 800-53 recommendations to limit the scope of the key—specifically, the AMK is only used to wrap/unwrap master keys, and master keys are only used to derive micro-partition file keys. In addition, Snowflake limits the scope of micro-partition file keys to 16 MB of compressed data only. To limit the lifetime of the key, Snowflake automatically—and by default—rotates the account master and the object

master keys in the hierarchy [once every 30 days](#). This means that new data is encrypted with a new key every 30 days. In addition, customers using enterprise or higher editions can choose to enable [annual re-keying](#), which rekeys the file keys that are used to decrypt micro-partitions that have not been changed for more than a year. The above key rotation capabilities are transparent to the end customers without any downtime.

In summary: The above keys are all created, maintained, and managed by Snowflake. Another encryption key management option is the hybrid Hold Your Own Key (HYOK) and Bring Your Own Key (BYOK) solution—or, in Snowflake parlance, Tri-Secret Secure.

Snowflake Tri-Secret Secure (TSS)

Snowflake Tri-Secret Secure (TSS)—the Snowflake hybrid Hold Your Own Key (HYOK) and Bring Your Own Key (BYOK) solution—lets the customer introduce their own customer-managed key (CMK) to the key hierarchy. Snowflake uses the customer-managed key to generate and protect an account master key called the AMK-C (Account Master Key–Customer). The CMK never leaves the customer's cloud key management system. Snowflake uses [HMAC](#) to combine the AMK-C and the AMK-S (Account Master Key–Snowflake) to generate a new composed key that wraps/unwraps the object master keys, as shown in figure 6.

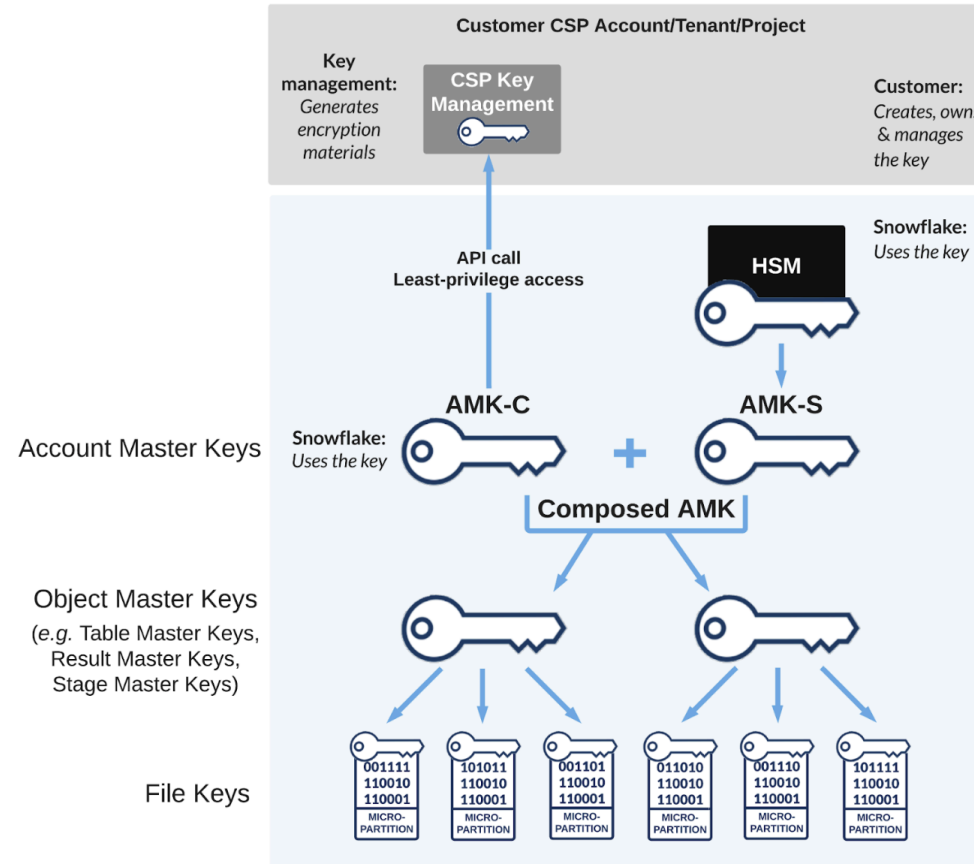


Figure 6. Snowflake TSS, aka "Bring Your Own Key"

Snowflake supports each cloud service providers' native key management system: Azure Key Vault, AWS KMS, and GCP KMS. The KMS, which is located in the customer's tenant/account/project, stores the customer-managed keys (CMKs). The customer then grants Snowflake access to their CMK using a well-defined, least-privilege access policy. Snowflake technical support generates the access policy document and shares it with the customer to either apply to their key, or to apply at the key-vault level, depending on the cloud service provider in use.

Snowflake TSS does not support custom hardware security modules (HSM) hosted in the customer environment. The customer can, however, generate a key using their own HSM, and then import the key into the AWS, Azure, or GCP key management systems. Snowflake can then use that imported key with TSS to generate the new account-composed key.

At any time, the customer can disable or delete the key, or revoke the key access policy. If that action is taken, TSS has been designed so that all data encrypted by TSS shall be encrypted to Snowflake within ten minutes, causing running queries to fail, and rendering data unreadable until the customer provides access to the same CMK again.

Note: Snowflake recommends enabling logging and monitoring on the KMS, and reviewing KMS access events to spot any unauthorized access.

In summary: The customer has control over their key in their trusted cloud provider. In addition to TSS, the customer can also identify sensitive/personal data that needs extra protection and use Snowflake column-level security with Snowflake `encrypt()/encrypt_raw()` or External Tokenization to encrypt/tokenize the sensitive data in those columns with their own key on top of TSS/BYOK. This option is discussed next.

Protecting specific columns with Snowflake column-level security

This section describes an additional layer of encryption/tokenization at the table's column level. So far we have discussed micro-partition file encryption and its encryption key hierarchy. With column-level encryption, the customer can protect specific columns using:

- Snowflake [Built-in functions](#)
- Snowflake [dynamic data masking policies \(DDM\)](#) with Snowflake `encrypt()/encrypt_raw()` encryption functions
- Bespoke UDFs that the customer maintains, or
- [External Tokenization](#) implemented by customer

The customer can define one DDM policy and apply it to multiple tables and multiple columns, or the customer can tag the sensitive columns and use [tag-based policies](#) to apply the extra protection required for them.

This capability can be implemented along with [classification](#) and [tagging](#), whereby the customer first classifies, then tags, sensitive data prior to applying the policies. With [tag-based policies](#) especially, customers can define column-level

encryption/tokenization policies at the tag level, and then once the tag is assigned to a sensitive column, the policies will automatically apply. This minimizes configuration errors and simplifies the protection of sensitive data.

The encryption keys used in `encrypt()`/`encrypt_raw()` and their decrypt counterparts, or any bespoke approach, are outside of the key hierarchy we discussed so far. The customer can provide encryption keys at query time, either manually or programmatically, using the external function capability in Snowflake as shown in figure 7. With this option, the external function retrieves the encryption key from the customer HSM, either on-prem or in the cloud. Either will work as long as the HSM supports API calls.

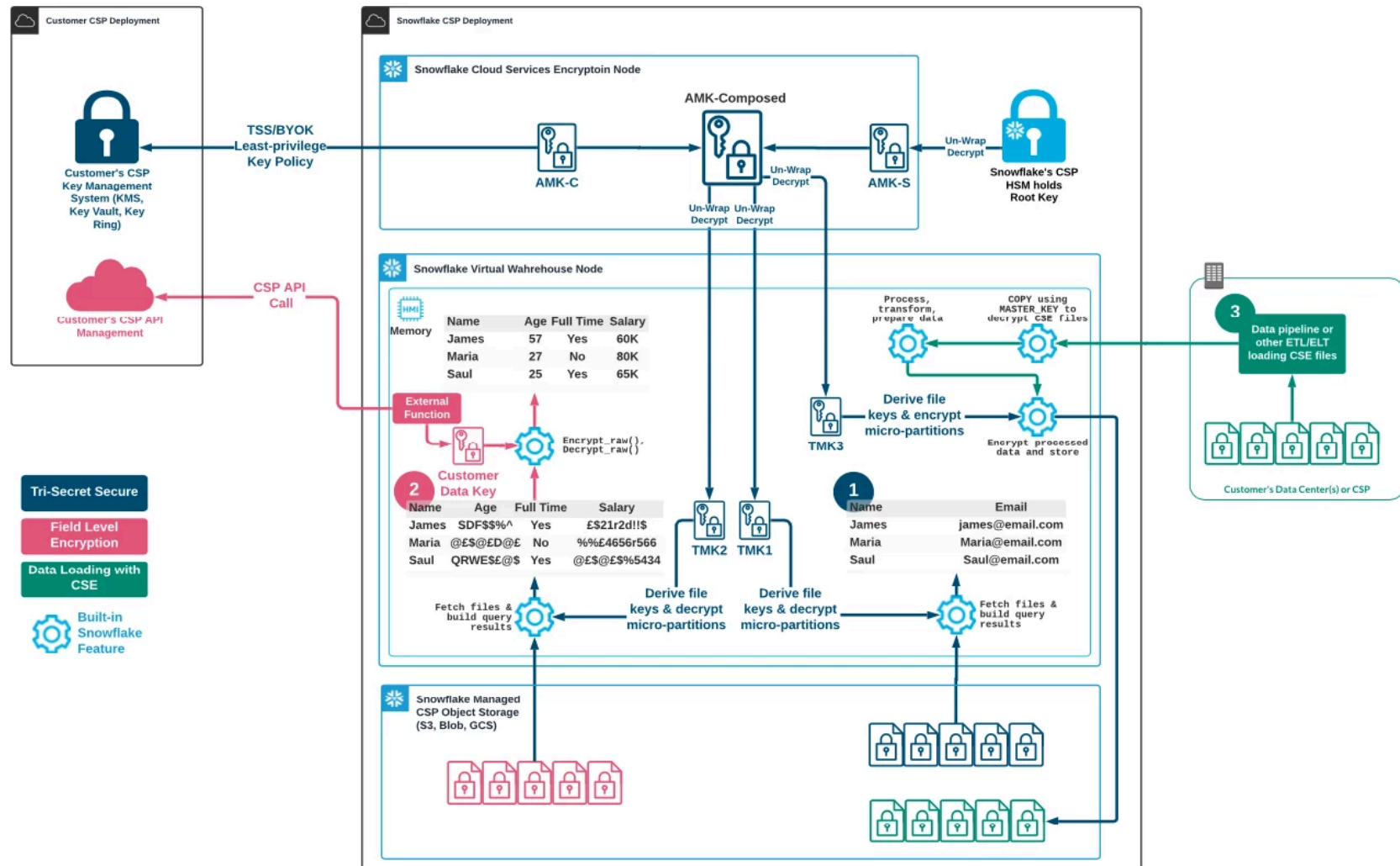


Figure 7. Encryption keys can be provided programmatically using Snowflake external function capabilities

Figure 7 shows two use cases:

- The customer data is encrypted at-rest using Snowflake TSS/BYOK as [discussed previously](#)
- Customer-specific columns are encrypted using a Snowflake dynamic data masking (DDM) policy with `encrypt()/encrypt_raw()` using HYOK on top of TSS/BYOK

In the following flow, assume an authenticated user with the proper [RBAC](#) role runs a query:

1. The user submits a query that needs to access a table with sensitive data.
2. The system loads the encrypted micro-partitions into the memory of the virtual warehouse.
3. In the memory of the virtual warehouse, the system uses the TSS/BYOK key hierarchy to decrypt the micro-partitions. The output is the requested table with encrypted columns.
4. If the user role allows the clear text column to be seen, the DDM kicks in and tries to decrypt the columns. DDM can invoke:
 - The `decrypt()/decrypt_raw()` built-in functions based on the AES encryption algorithm with a key provided programmatically via external functions
 - Or: The [detokenization function](#) and use the external tokenizer to detokenize/re-identify the sensitive data in that particular column
 - Or: A UDF written in [Java](#) or [Python](#) that uses additional encryption techniques, like format preserving encryptions
 - Or: Masking or partial masking of the data at run time

In the case of `decrypt()/decrypt_raw()`, or a customer Java/Python UDF, the customer can provide the key manually as part of the dynamic data masking policy (not recommended), or—better—the customer can use the [external functions capability](#) (figure 7), which calls the customer API management/API gateway and retrieves the encryption materials from the customer key management system programmatically. The customer key management system can be a cloud key-management system, a cloud-dedicated HSM, or even an on-premises HSM.

Once the external function successfully returns the encryption key materials, the key is passed via the tag-based or column-level policies to the system to decrypt/detokenize the data—but only if the authenticated user has the proper role and conditions to see the encrypted data.

Figure 8 shows how the customer can combine [row access policies](#) with [column-level security](#) to restrict records based on a specific condition, and then pass the results to be decrypted by the column-level security policy.

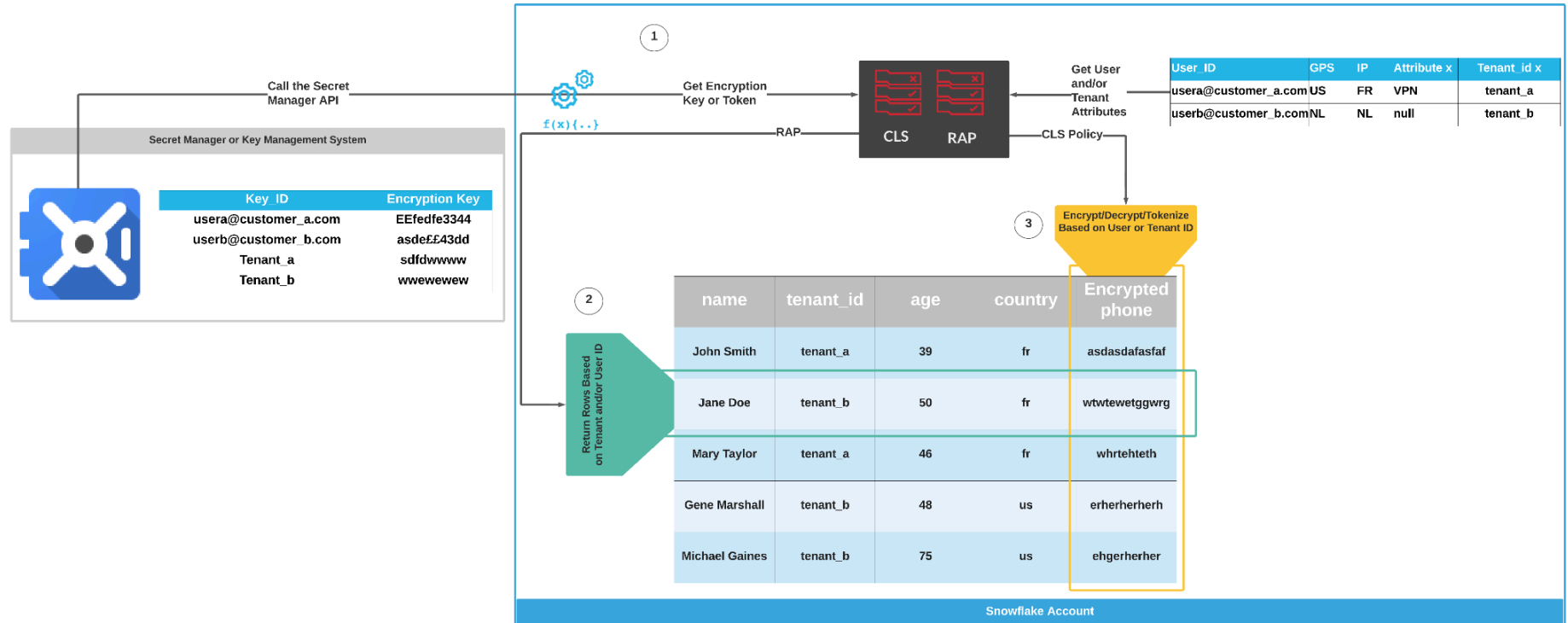


Figure 8. Using both RAP and CLS to external protect the sensitive columns

In summary: With Snowflake column-level security, the customer can add an extra layer of protection (including encryption, tokenization, or masking) on top of Snowflake TSS/BYOK. This extra layer of protection uses the customer key only at query time when it is needed, and the customer HSM provides the key only when it is needed by an authorized user. With this option, the protected columns are triple-encrypted at rest: at the cloud storage level, at the micro-partition level with TSS, and at the column level with CLS.

When using [CLS](#) or [External Tokenization](#), the query results are first staged in the Snowflake Internal Stage, where they are encrypted using [Snowflake Default out-of-the-Box Encryption](#) or [Snowflake Tri-Secret Secure \(TSS\)](#). This is temporary storage that is only used to transparently return the results back to the requested entities. Customers with strict requirements that require that even temporary results be encrypted in memory in the virtual warehouse when the results are staged, however, can choose a Snowflake [partner](#) who offers inline gateways. That topic is discussed in the next section.

Split processing or multi-party processing

[Split Processing](#) or [Multi-Party Processing](#) encrypts sensitive data so that it cannot be reconstructed or re-identified by any single data processor, such as Snowflake. Only the customer can reconstruct the data by accessing the following resources in combination:

- A. A local repository or an HSM that resides in the customer environment, and which contains the keys/tokens necessary to re-identify/reconstruct the data.
- B. The data in the customer's Snowflake account, which has sensitive data columns protected by external systems that the Snowflake account cannot access to retrieve the information needed to re-identify or reconstruct the data, such as encryption keys, tokens, pseudonymization local repositories, and so on.
- C. An external system that can use A+B to reconstruct/re-identify sensitive data, such as a Snowflake [security partner](#) or another [Snowflake account](#).

Hold Your Own Key (HYOK) in the customer environment

Customers choose this option because they decided that no encryption keys will be provided to their Snowflake account to decrypt any sensitive data at query time to return the results. In this scenario, the customer wants query results with sensitive data to be encrypted in memory in the Snowflake virtual warehouse and in the staging storage. The customer then uses their policy enforcement point (PEP) solution in their environment to encrypt/tokenize/pseudoanonymize the sensitive data before it leaves their environment.

Snowflake supports this scenario with the help of [security partners](#) who provide data discovery, data classification, data encryption, anonymization, and tokenization services for customer data in the middle, before it is loaded and ingested into Snowflake via a PEP solution. This means that the customer classifies data, and then chooses to protect the sensitive columns before it leaves their environment using their own HSM and/or local repositories so that those columns remain protected both in memory while processing, as well as encrypted at rest in the customer's Snowflake account. The customer's Snowflake account does not have access to the required encryption keys/tokens.

The customer has full control of the PEP, which may live in the customer environment, or in another SaaS, and the PEP can only access the customer HSM to retrieve the key and the identification data to decrypt/re-identify the query results as shown in figure 9.

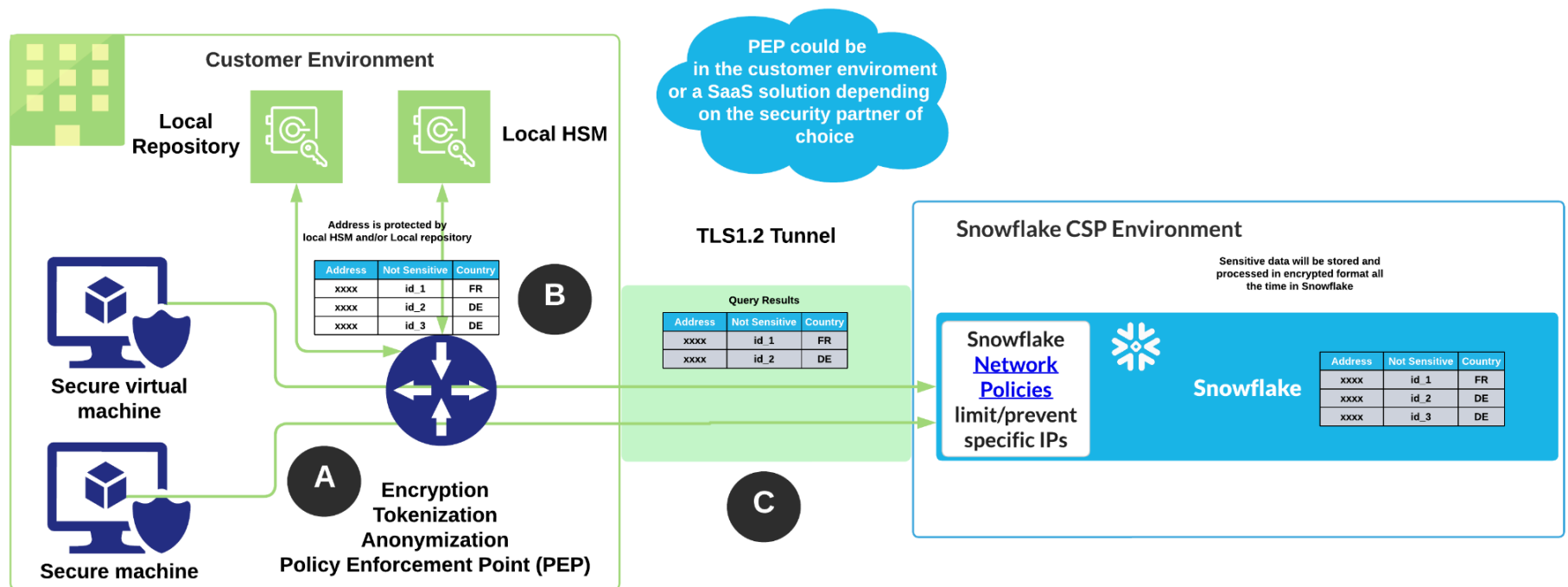


Figure 9. Split processing, anonymization, tokenization, and HOYK

This solution works as follows:

- **A:** The customer classifies and tags the sensitive data that needs extra protection.
- **B:** The PEP encrypts/tokenizes/anonymizes the sensitive data according to the customer's policies before it leaves the customer's environment, and then uses Snowflake drivers and connectors to store the data in a protected format in Snowflake. The ingested sensitive data remains in the protected format both in memory and while in Snowflake storage.
- **C:** When the client runs a query, the PEP processes it and sends it to Snowflake, where dedicated compute nodes in Snowflake (aka virtual warehouses) process the query. Sensitive columns are encrypted/tokenized/anonymized by the PEP. Snowflake virtual warehouses return the results over a TLS 1.2 tunnel to the PEP, and only the PEP in the customer environment is able to decrypt/re-identify the sensitive data using the customer's local HSM/local repository before delivering it back to the clients.

This option is for customers who have strict requirements that sensitive data cannot be decrypted, de-tokenized, and/or de-anonymized anywhere outside of the customer environment. Therefore, customers who choose to implement such a solution are satisfying the [multi-party processing](#) (also known as [split processing](#)) requirements where neither Snowflake alone nor the partner alone can reconstruct or re-identify the sensitive information. Only the customer, who is the only party who has access to their Snowflake account, the PEP, and their local repository/HSM, can reconstruct/re-identify the sensitive data.

Secure data sharing

The concept of multi-party processing or split processing can be implemented natively using [Snowflake Data Clean Rooms](#). Clean Rooms are powered by Snowflake Secure Data Sharing technology, and each data provider can apply the appropriate [Security of Data Processing Measures](#) at the source to make sure only cleansed, sanitized, and approved data is shared in real time.

Therefore, each data provider is a data processing party by itself, and the data aggregator is the only one with a wider view of all the data sources. But it does not stop there—the data aggregator can only see what each data provider allows them to see based on each party's business, regulatory, and compliance requirements.

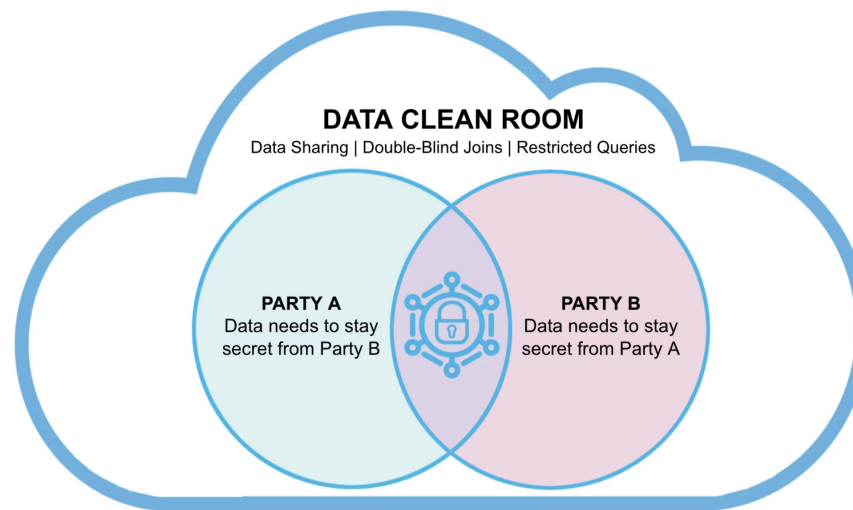


Figure 10. Data Clean Room

Data Pseudonymization

Pseudonymization is the process by which personally identifiable information is replaced by one or more artificial identifiers or pseudonyms. If a customer needs to pseudonymize data before loading it into Snowflake, one option to consider is anonymizing the data in the ingestion pipeline using an external tool. Suitable tools are discussed in the [Snowflake column-level security and Customer Data Key](#) topic, and the [Hold Your Own Key \(HYOK\) in the customer environment](#) topic, where, instead of encryption, the customer will be anonymizing the data itself.

Note: Snowflake supports [data anonymization](#) out-of-the-box. This anonymization creates anonymized views that leverage the k-anonymity algorithm on the top of the data stored in tables, which means that it does not anonymize the original data. The consumers of the anonymized views will not be able to re-identify the data because the identifiers have already been removed from the view.

Summary of encryption and anonymization solutions

The following table summarizes the different options to protect data processing

The Data Protection Option	Snowflake Default out-of-the-Box Encryption	Snowflake Tri-Secret Secure (TSS)	Snowflake column-level security and Customer Data Key	Split Processing – Hold Your Own Key (HYOK) in the customer environment	Split Processing – Secure Data Sharing
Protection keys/token ownership	Snowflake	Customer	Customer	Customer	Depends on each Snowflake account setting*
Protection Key/Token management	Snowflake	Customer	Customer	Customer	Depends on each Snowflake account setting*
Protection Key/Token usage	Snowflake	Snowflake	Snowflake	The customer in their environment(s), only	Depends on each Snowflake account setting*
Sensitive Data	Decrypted in	Decrypted in	Decrypted in	Sensitive columns are	Depends on each Snowflake

Columns Processing	memory in the customer's virtual warehouses	memory in the customer's virtual warehouses	memory in the customer's virtual warehouses	processed encrypted, even in the memory of the virtual warehouses	account setting*
Legal or government request	LAW ENFORCEMENT REQUESTS	LAW ENFORCEMENT REQUESTS	LAW ENFORCEMENT REQUESTS + Customers can set up their own auditing and monitoring	LAW ENFORCEMENT REQUESTS + Snowflake alone has no access to the encryption keys/tokens	LAW ENFORCEMENT REQUESTS + Depends on each Snowflake account setting*
Customer ability to lock data access in case of legal or government request	None	(Reactive) Customer can revoke access to their CMK	(Reactive) Customer can revoke access to their CMK and/or revoke access to the column-level encryption key at run time	(Preventive) Snowflake does not have access to the encryption keys/tokens that Snowflake uses to process sensitive columns in protected format	Depends on each Snowflake account setting*

Note: Customers can choose to implement any of the other options when they use [Secure Data Sharing](#) and data clean room.

B. Data confidentiality, integrity, availability, and the resilience of data processing systems

Customer data processing confidentiality and integrity

We covered data confidentiality in the [Encryption and pseudonymization](#) section. When it comes to data integrity, customer data is stored in encrypted micro-partitions. These are write-once, read-many files, as mentioned in the [Snowflake default out-of-the-box encryption](#) section. In the event the customer wants to change a record in a file, the system creates a new file and keeps the older version of the file for a period of time equivalent to the [Time Travel](#) setting

(which is configurable from 0–90 days), plus the [Fail-safe time](#), which is a fixed, non configurable, seven days. In addition, the file keys (FK) that are used to encrypt/decrypt the micro-partitions are derived from the object master keys (OMK), plus the metadata of the file that preserves the integrity of the micro-partitions.

High availability

This section discusses high availability and data processing system resilience.

Regional availability and resilient customer data processing

Snowflake is built on the infrastructure of three cloud service providers (AWS, GCP, and Azure). Each Snowflake region is deployed over a minimum of three availability zones. In addition, the Snowflake architecture separates cloud services, from compute, and storage, so services, compute, and storage can scale up and down independently as shown in figure 11.

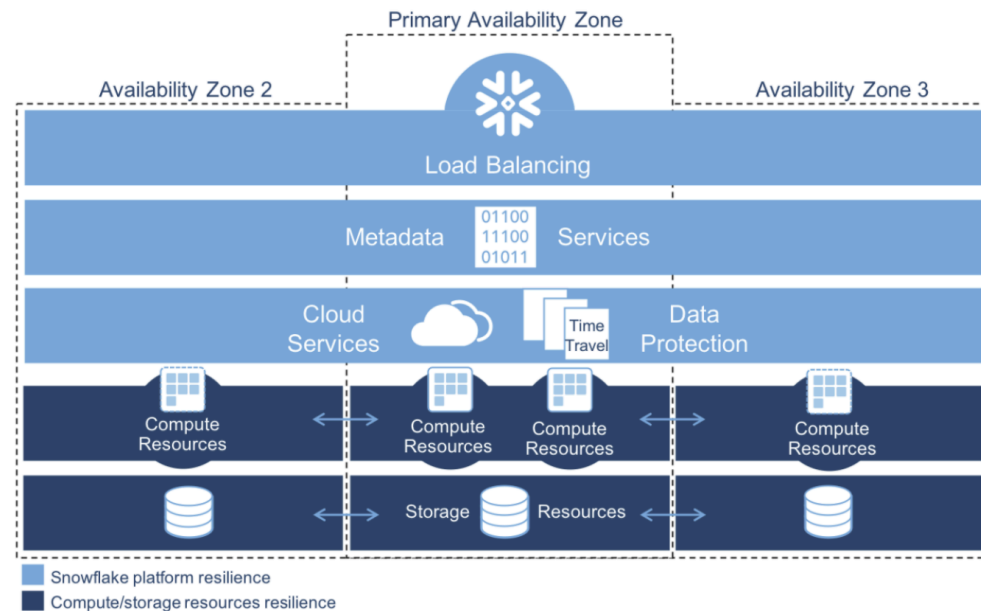


Figure 11. The Snowflake high availability architecture

In summary, Snowflake provides:

1. Storage high availability and resiliency by replicating across a minimum of three availability zones.
2. Compute high availability and query resilience by leveraging compute in a minimum of three availability zones. The load balancing layer transparently redirects the load to an available zone in case of a zone failure.
3. Service layer high availability and resiliency by the same means as in (1) and (2).

For more details, see [How to Make Data Protection and High Availability for Analytics Fast and Easy](#) on snowflake.com.

Beyond standard cloud data protection, Snowflake provides the following features that increase data integrity, high availability, and resiliency:

- [Time Travel](#) enables the customer to recover data from any point, up to 90 days prior. In addition, it's all accomplished automatically. Other than specifying the number of days for Time Travel retention at setup (default is 24 hours for Snowflake Enterprise and above), the customer does not have to initiate a thing or manage snapshots.

Note: Time Travel impacts the right to be deleted. For details, see the [Right to Be Forgotten](#) section.

- [Fail-safe](#), which is separate and distinct from [Time Travel](#), ensures that historical data is protected in the event of a system failure or some other event, such as a security breach or if the customer disables Time Travel and then accidentally deletes or corrupts their data. Fail-safe provides a (non-configurable) seven day period during which historical data may be recoverable by Snowflake. This period starts immediately after the Time Travel retention period ends. Note, however, that a long-running Time Travel query will delay moving any data and objects (tables, schemas, and databases) in the account into Fail-safe, until the query completes.

Note: Fail-safe impacts the right to be deleted. For details, see the [Right to Be Forgotten](#) section.

Cross-cloud, cross-region availability and customer data processing resilience

The [regional availability and resilient customer data processing](#) topic discusses the Snowflake platform's high availability and resiliency in a single cloud region. Some customers may have additional cross-region, and even cross-cloud, high

availability and resilience requirements. For such customers, Snowflake provides a [cross-region and cross-cloud replication and failover](#) capability that helps customer with:

- [Database Replication](#): To replicate their databases across regions, customers can be very specific as to which regions and geography they would like to replicate their data to. For example, customers can replicate their datasets across two different regions within the European Union.
- [Account Replication](#): Extends the replication capability to Snowflake account [level objects](#), such as:
 - Network policies
 - SAML integrations
 - Roles and Users
 - SCIM Integrations
 - Shares and Warehouses

This minimizes the high availability setup and makes it as easy as possible for customers.

- **Failover/ Failback**: In case of complete region failure, the customer can initiate failover to an active region in the same or a different cloud provider according to their business continuity plans.
- [Redirecting Client Connections](#): Client redirect enables redirecting the customer's client connections to Snowflake accounts in different regions for business continuity and disaster recovery, or if the customer is migrating their account to another region or cloud platform. This capability provides customer applications with one single connection to Snowflake across multiple regions and makes failover transparent to those applications.

Note: See [Supported Snowflake Cloud Regions](#).

Figure 12 shows different failure scenarios and how Snowflake helps to increase data processing availability and resiliency.






FAILURE		MITIGATION
	Customer Error	Snowflake Features Time Travel Fail-safe
	Single Instance Failure	Snowflake Built-in Redundancy Triple-redundancy for critical services Automatic retries for failed parts of a query
	Zone Failure	Snowflake Built-in Redundancy Using Availability Zones on AWS, Azure, GCP Using Availability Sets on Azure
	Region Failure	Snowflake Features Cross-Region Replication Cross-Region Failover
	Multi-Region Failure	Snowflake Features Cross-Cloud Replication Cross-Cloud Failover

Figure 12. Possible failure scenarios and how Snowflake mitigates them

Figure 13 shows how account and database replication, and client redirect helps in failover/failback scenarios.

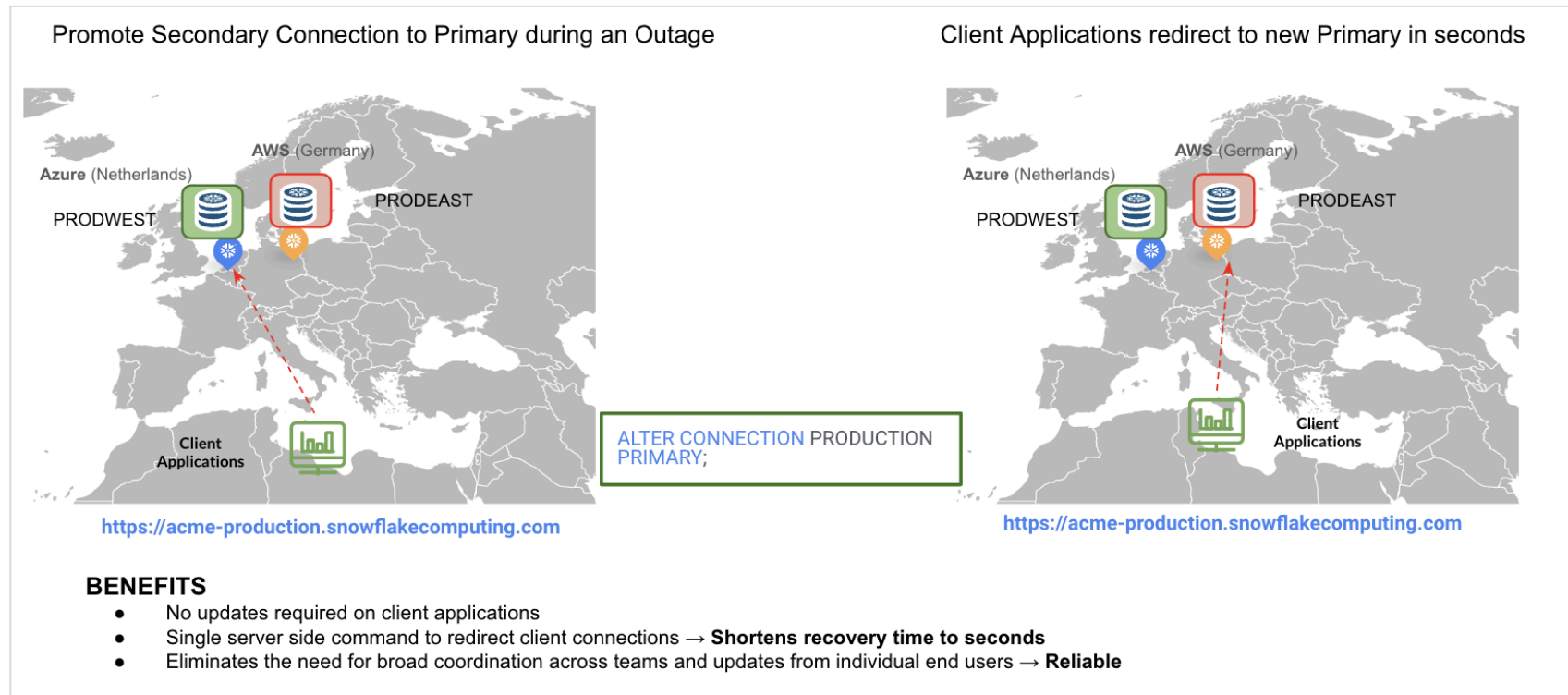


Figure 13. Snowflake cross-region and cross-cloud data processing high availability and resilience

Off Snowflake backup options

In addition to regional high availability, Time Travel, Fail-safe, replication, and client redirect, customers have the option to [unload their data](#) in a [standard format](#), such as CSV, Parquet, JSON, and so on, to their cloud provider's storage buckets or on-prem storage.

This provides the following benefits to the customer:

- No vendor lock-in. Snowflake exports data in standard formats, which makes it easier for customers to ingest into other platforms.

- Offline storage of a copy of their sensitive data. Customers need to design an unloading process that runs on their desired schedule to make sure that their backups are current according to their business continuity plans.

The following table shows the available unloading options.

Unloading Option	Descriptions	Security Considerations
External Stage	<p>This can be customer owned and managed, or AWS S3, GCP GCS, Azure Blob, or any S3-compatible storage (on premises or edge). In the case of cloud storage services, the customer may choose to offload sensitive data to be stored on premises.</p>	<ul style="list-style-type: none"> • All Snowflake-to-the-cloud service communications are encrypted in transit using TLS 1.2. (See Data encryption in-transit.) • The customer owns the storage service(s) and must restrict access to only authorized identities (using least-privilege access principles). • The customer should restrict access to the External Stage to authorized networks only via S3 policies, Azure Storage firewall, and GCS policies. • For sensitive data, the customer should consider implementing client-side encryption in addition to the default server-side encryption. • In the case of client-side encryption, the customer's responsibility is to manage and maintain the encryption keys. • Customers should enable auditing and monitoring of their cloud storage and integrate that with their security operation center along with Snowflake Account Usage (Copy History) to monitor any unauthorized access to their storage services.
Using Snowflake Drivers and Connectors	<p>The customer can use machines in the cloud or on-prem that run Snowflake drivers or connectors.</p> <p>With this option, the customer can load data directly to the cloud or on-prem without an intermediate External Stage.</p>	<ul style="list-style-type: none"> • Snowflake uses TLS1.2 to encrypt customer data between the driver/connector and the service (see Data Encryption In-transit). • Snowflake leverages client-side encryption by default. Use GET/PUT operations with the drivers and connectors. • Encryption keys for the client-side encryption are managed automatically via the Snowflake encryption key hierarchy using the Stage Master Key as described in the Snowflake default out-of-the-box encryption section. • Customers can use private connectivity between the drivers and connectors and Snowflake services (see Data Encryption In-transit).

C. The ability to restore access to sensitive data after failure

The [Data confidentiality, integrity, availability, and resilience of data processing systems](#) section presents the options that customers can adopt to build their data resiliency, high availability strategy. In this section, we compare the options to help customers choose the best fit for their processes.

Considerations	Regional HA	Cross Regions/CSPs HA	Off Snowflake
Setup needed	No, all Snowflake accounts are regionally available	Yes, built in. Customers must configure account replication with or without client redirection .	Yes. Customers must set up and maintain their external storage system (CSP or on-prem) and manage their backup frequency according to their needs.
Network Policies	Yes, Snowflake network policies	Yes, Snowflake network policies	Customers must maintain the storage policies separately as per the CSP or on-prem storage capabilities.
IAM	Snowflake authentication and authorization	Snowflake authentication and authorization	The customers must maintain the storage IAM policies as per the CSP or on-prem storage capabilities.
Data Governance	Snowflake single pane of glass	Snowflake single pane of glass	The customer must make sure they have the proper data protection, classification and protection is configured separately as per the CSP or on-prem storage capabilities.
Encryption	Snowflake encryption options	Snowflake replication encryption and Snowflake encryption options	Customers must maintain the same protection such as masking, anonymization, encryption as per the CSP or on-prem storage capabilities.

Auditing and Monitoring	Snowflake auditing and monitoring	Snowflake auditing and monitoring	Customers maintain external storage auditing and monitoring as per the CSP or on-prem storage capabilities.
Restore After Failure	<p>In case of availability zone failure, customer error, or single compute instance failure: RPO=0 RTO=0</p> <p>In case of regional failure and without replication: The Snowflake environment will be available within 120 hours of the cloud region availability.</p>	<p>In case of regional or cloud failure.</p> <p>With client redirect, RTO is as long as it takes to promote the secondary region.</p> <p>If PrivateLink/Private Link or automatic failover is needed, please contact Snowflake.</p> <p>RPO depends on the frequency of updates on the primary and the frequency of replication could be a minimum of 1 min.</p>	<p>Customers have two options:</p> <p>Option 1: Use an external table temporarily while executing option 2 if needed</p> <ul style="list-style-type: none"> • RTO = Customers can run queries against the external tables. • RPO = Depends on the frequency of data unloading to the external storage. <p>Option 2: Restore and re-ingest the data in Snowflake after failure.</p> <ul style="list-style-type: none"> • RTO = Depends on the customer restore process and the size of the data. • RPO = Depends on the frequency of data unloading to the external storage.

D. Regularity testing and evaluating data processing security

This section covers Snowflake from two different points of views: (1) The Security of the Snowflake platform as a service, and (2) The security of customer data that the customer ingests into their Snowflake accounts (instances).

Regularity testing and evaluating the security of the Snowflake platform as a service

Snowflake is a self-managed service. Accordingly, Snowflake maintains the security of the platform from the backend according to industry standards. Further, Snowflake undergoes regular testing and evaluations as shown in figure 14. For details, please visit the [Snowflake Security and Trust Center](#). All relevant documentation and reports are available upon request.

Note: Some reports are distributed under NDA, such as SOC2, DRP, BCP, and so on. Other documents are available online, such as [Snowflake security policies](#), [Data Processing Addendum](#), [Terms of Service](#), and [Snowflake ISO 27001 Certificate](#).

INFOSEC & COMPLIANCE

at a Glance


All reports, attestations, documentation, and certifications

Third-Party Reports & Certifications

- Snowflake SOC 1 and 2 Type II Report
- Snowflake PCI-DSS-AOC-Final Report
- HITRUST Certification w/ HIPAA scoping factor (with Shared Responsibility Matrix)
- Snowflake's ISO 27001 Certificate (includes ISO/IEC 27017 & ISO/IEC 27018)
- ISO 9001
- CSA STAR Level I
- BSI C5 (Germany)
- FedRAMP (Package on OMB MAX)
 - FedRAMP [High](#) (AWS), [Moderate](#) (Azure)
 - DoD CC SRG IL4
- StateRAMP & TX-RAMP
- IRAP Protected
- Cyber Essentials Plus
- CyberGRX Report
- Penetration Test Results



ISO/IEC 27001
ISO/IEC 27017
ISO/IEC 27018




SOC 2 Type II
12 Month Coverage Period
SOC 1 Type II
6 Month Coverage Period



DoD CC SRG IL4:
AWS GovCloud



AGID Agenzia per l'Italia Digitale




PCI-DSS



Moderate and High on
SnowGov regions:
AWS, Azure



BSI C5



INHERITANCE PROVIDER
HITRUST
i2 CERTIFIED



CYBER ESSENTIALS PLUS




800-171




IRAP




NHS
nhs.uk



AUTHORIZED
StateRAMP



TX-RAMP
CERTIFIED



STAR
LEVEL ONE



ITAR



HM Government
G-Cloud 13
Supplier



TISAX

Snowflake's Policy Documentation

- [Snowflake Security Addendum](#)
- <https://www.snowflake.com/legal/> for Acceptable Use, Support, and more
- [Latest Certifications and Reports](#)

Snowflake Internal Controls & Testing

- DRP, BCP, and Information System Contingency Plans
- Security Incident Process
- Staff Training, Onboarding, and Access Policies

Snowflake Self-Assessment Reports

- CAIQ
- SIG Lite
- Red Team Pen Tests

Figure 14. Snowflake Platform Security

Figure 15 shows the 24x7 security monitoring, the regular third-party penetration testing, and the vulnerability scans that Snowflake conducts.

© 2024 Snowflake Inc. All Rights Reserved

43

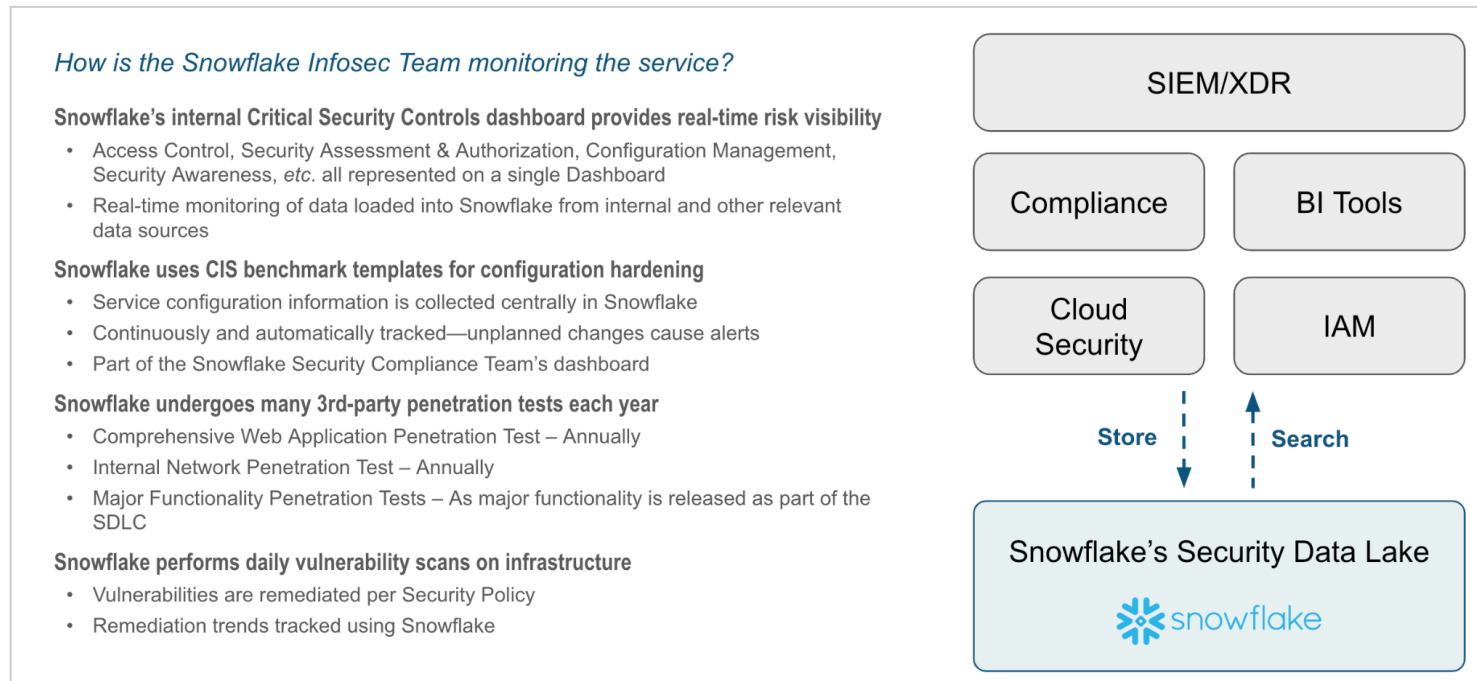


Figure 15. Snowflake infrastructure security and monitoring

Regularity testing and evaluating the security of customer data

Customers are responsible for regularly testing and evaluating the security of their data. Snowflake provides its customers with a wide set of [data security capabilities](#) and [data governance capabilities](#) that customers can implement to develop a process to regularly test and evaluate the effectiveness of their security controls.

Snowflake logs all user interactions with Snowflake in a set of [history objects](#) (Account_Usage) that customers can access and use in multiple ways, as shown in figure 16.

Customers can integrate their Snowflake accounts with their [SIEM](#), [SOAR](#) (Security Orchestration, Automation, & Response), or [XDR \(extended detection and response\)](#) tools via Snowflake [drivers or connectors](#). Integration allows auditing and monitoring tools direct access to the Snowflake history objects. Alternatively, customers can export audit logs to a cloud storage service, and these tools can ingest them.

The following history objects are discussed next:

- [Login History](#)
- [Query History](#)
- [Object Dependencies](#)
- [Access History](#)

Note: Snowflake has a security features checklist that includes sample queries to help customers get started with Snowflake security monitoring. Customers should work with their Snowflake point of contact to request the latest copy of the security features checklist. Snowflake customers can also easily build a [security health dashboard](#).

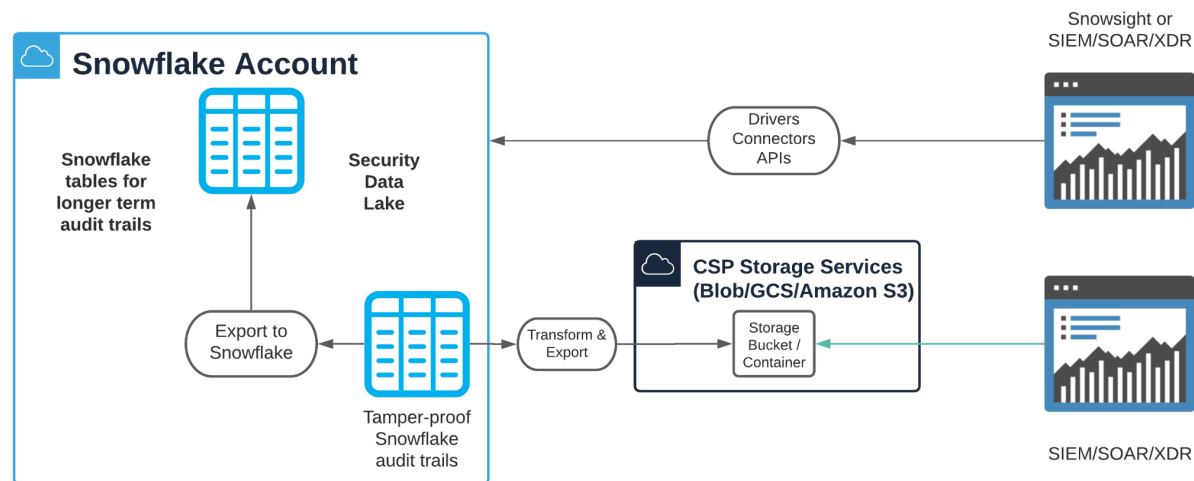


Figure 16. Integrating Snowflake with auditing and monitoring solutions

Note: Some Snowflake customers use Snowflake at the core of their security operation center as their security data lake. For more information on this topic, refer to [Snowflake for Cybersecurity](#).

Account usage vs. information_schema

Account usage history objects have [some latency](#), while [information schema](#) table functions do not. The application provider can create views to vertically join data based on event time stamps from both the account usage and information_schema data sources, and provide full, real-time visibility as shown in the following example, which combines the [login history](#) account usage view with the [login history information schema table function](#).

```
CREATE OR REPLACE VIEW GEOFENCING.PUBLIC.REAL_TIME_LOGIN_HISTORY AS
WITH cst_is AS
(
  SELECT event_id
    , event_timestamp
    , event_type
    , user_name
    , client_ip
    , reported_client_type
    , reported_client_version
    , first_authentication_factor
    , second_authentication_factor
    , is_success
    , error_code
    , error_message
    , related_event_id
    , 'Information Schema' record_source
  FROM TABLE(snowflake.information_schema.login_history())
)
, cst_au AS
(
  SELECT event_id
    , event_timestamp
    , event_type
    , user_name
    , client_ip
    , reported_client_type
    , reported_client_version
    , first_authentication_factor
    , second_authentication_factor
    , is_success
    , error_code
    , error_message
    , related_event_id
    , 'Account Usage' record_source
  FROM snowflake.account_usage.login_history
)
WHERE (event_timestamp, event_id) NOT IN (SELECT event_timestamp, event_id FROM cst_is)
```

```
)  
SELECT * FROM cst_is  
UNION ALL  
SELECT * FROM cst_au  
;
```

The [ACCOUNT_USAGE](#) data source has many views that store audit logs for various user activities with the Snowflake account. The ACCESS_HISTORY view, LOGIN_HISTORY view, and QUERY_HISTORY view sections that follow describe views that are interesting from a security perspective.

Login_History

The [LOGIN_HISTORY view](#) records Snowflake account customer user login attempts, and captures the time, login IP address, driver version, authentication method, and more. Further, by using a three-way JOIN between LOGIN_HISTORY, sessions, and QUERY_HISTORY, customers can determine how a user session that was used for a given query was authenticated.

Figure 17 shows an example of a dashboard built using LOGIN_HISTORY audit logs.

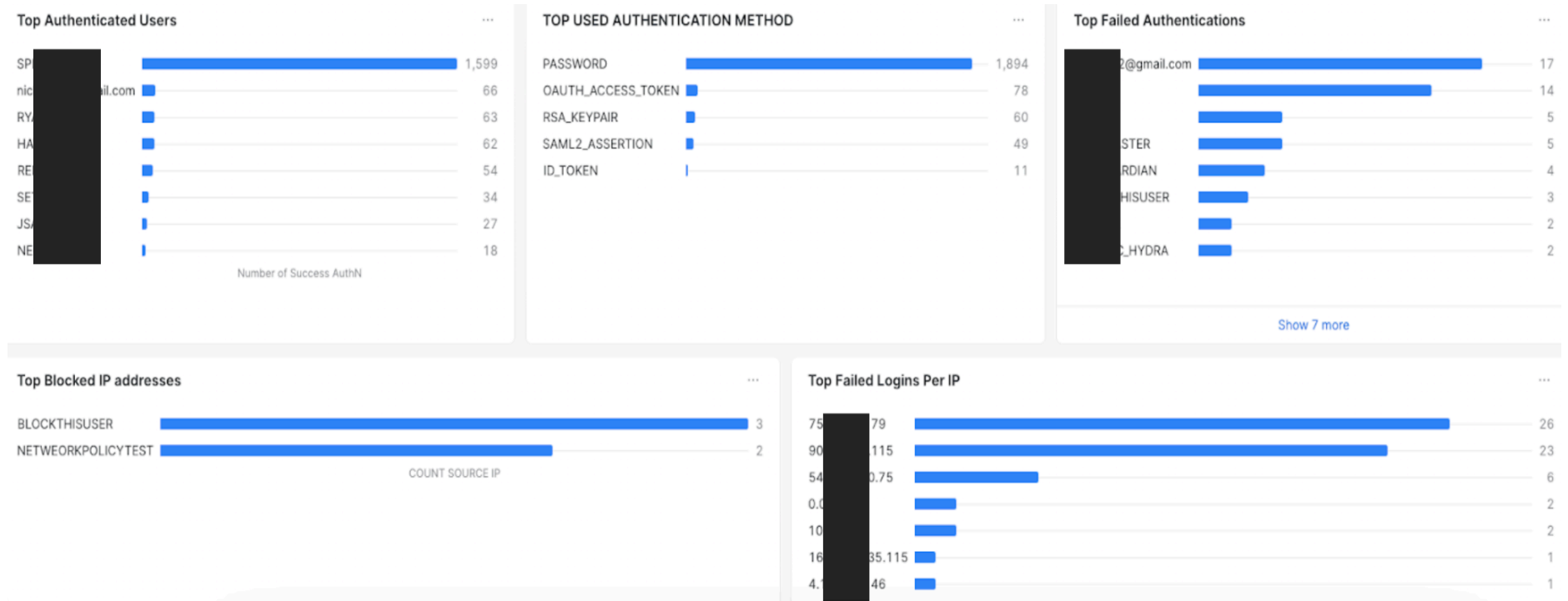


Figure 17. Dashboard built from data from the LOGIN_HISTORY audit logs

This JOIN shows who, when, and from where an entity accessed the datasets.

Anonymize personal data in account history

Snowflake provides its customers with security and confidentiality measures that customers can implement to protect personal data that is part of a customer's account usage. Customers can choose to obfuscate or anonymize those details in account history by providing surrogate identities to Snowflake as follows:

- A. Customers can provision users in Snowflake either manually or automatically. (See figure 18.)
 1. Manually: Customers can use SQL to provide surrogate identities in Snowflake, and maintain the mapping between real and the surrogate identities in their local repositories, as shown in [Hold Your Own Key \(HYOK\) in the customer environment](#).

2. Automatically: Using [SCIM](#), customers can set up their identity providers, such as Azure, Okta, Pinfiderate, and so on, to have custom mappings between real and surrogate identities. SCIM then provisions the surrogate identities in the customer's Snowflake accounts only. In this case, the local/third-party repository is the customer IdP.

In the case of option A, only the customer can re-identify the real user identities by using their IdP's user database or their local repository.

- B. To obfuscate or hide the end user's IP addresses, customers can implement one of the following options when connecting to Snowflake. (See figure 19.)
 1. [NAT](#): With network address translation, customers can hide end-user IP addresses so that the Snowflake service only sees the NAT gateway IP address.
 2. [Proxy](#): Works the same as NAT with regards to hiding real IP addresses.
 3. [Private IP addresses](#): Customers can choose to connect to Snowflake via a [private connectivity option](#). In this case, Snowflake services only see the private IP addresses that are not globally unique. Customers can use this option with the NAT and Proxy options for an additional layer of masking. Private IP addresses are deemed personal and sensitive according to the customer's interpretation.

In the case of option B, only the customer in possession of their NAT, Proxy, and/or Private IP addresses database can map specific IP addresses to a specific person.

In summary: By combining options A and B, only customers who use their own internal systems can map or re-identity the real user identity. Snowflake is unable to use account usage data to map a specific login or query to a specific real user identity.

Customers should consider integrating the above options with their SOC so that they can keep full visibility across their enterprise applications, including Snowflake.

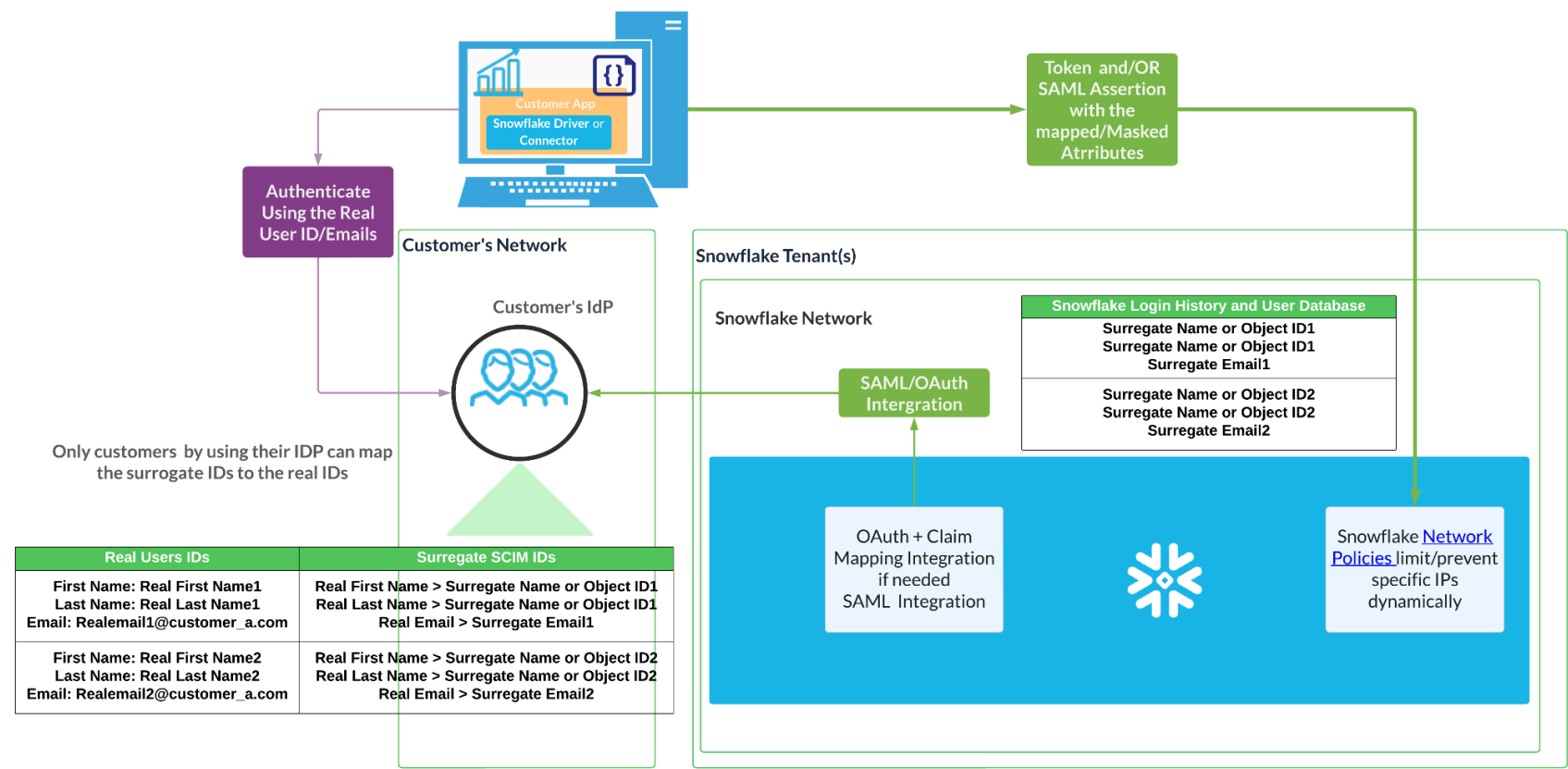


Figure 18. Custom SCIM mapping to hide the real user identities in Account_Usage

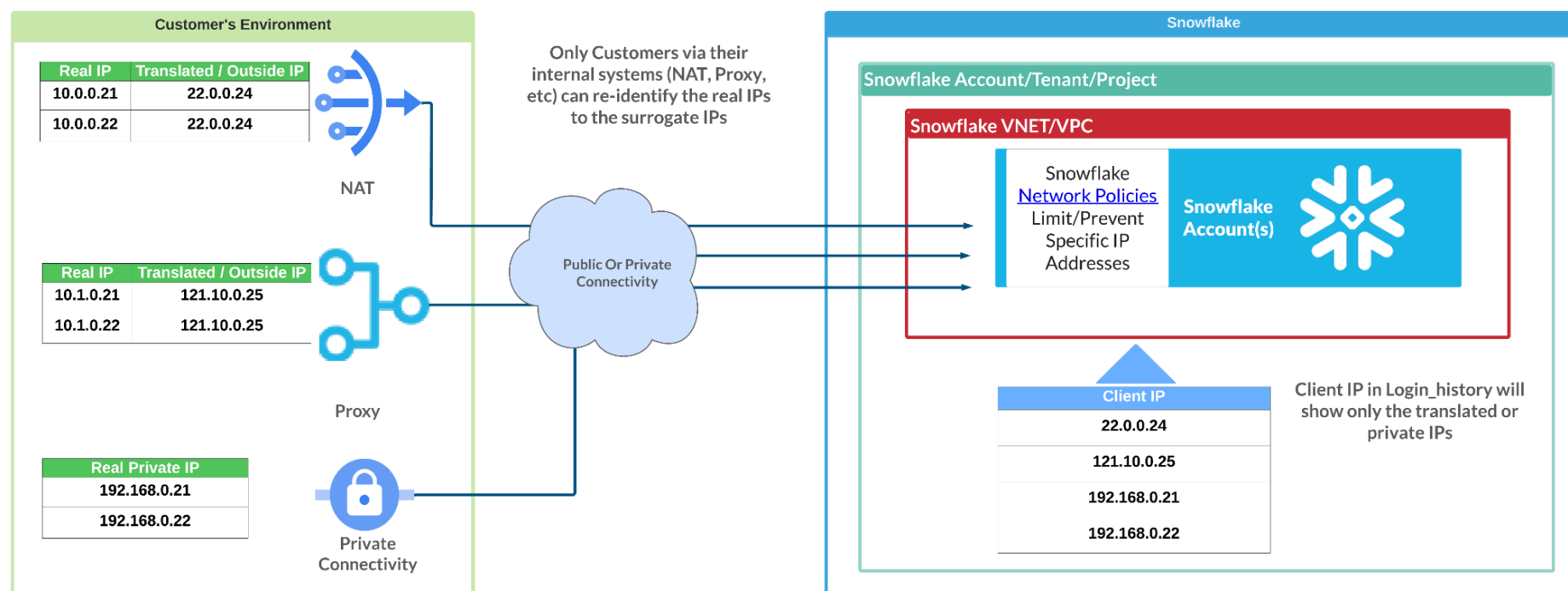


Figure 19. Hiding the real IP addresses

Query_History

The [QUERY HISTORY view](#) provides audit logs for queries executed in a Snowflake account, including time range, session, user, role, warehouse, and so on. For instance, ACCOUNTADMIN roles should not be used for daily account operations. Figure 20 shows an example of a custom Snowsight® dashboard that highlights that.

Why Are You Running as ACCOUNTADMIN? 9 rows ***

QUERY_TYPE	USER_NAME	START_TIME
SELECT	DN [REDACTED]	2021-06-14 02:31:53.282 -0700
SHOW	BO [REDACTED]	2021-04-05 05:47:59.189 -0700
SHOW	SE [REDACTED]	2021-02-24 08:58:16.539 -0800
SHOW	JS [REDACTED]	2020-11-19 08:15:10.216 -0800
ROLLBACK	WA [REDACTED] WILSON	2020-10-26 08:38:54.073 -0700
SELECT	SY [REDACTED]	2020-10-14 15:38:47.130 -0700
SHOW	JO [REDACTED]	2020-09-14 05:52:30.765 -0700
SHOW	EL [REDACTED]	2020-08-27 15:55:49.126 -0700
SHOW	RY [REDACTED]	2020-08-23 12:29:40.470 -0700

Figure 20. A Snowsight dashboard built from the QUERY_HISTORY view

Object dependencies

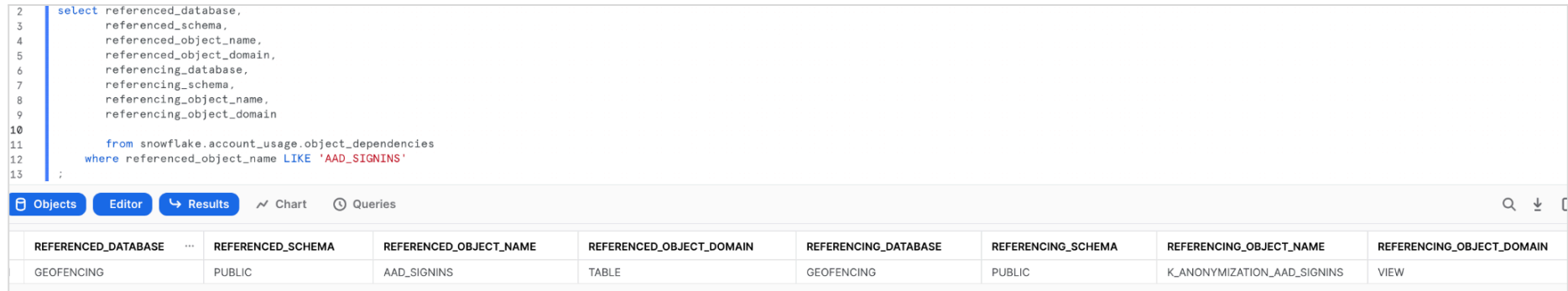
An [object dependency](#) means that, to operate on an object, the object being operated on must reference metadata for itself, or reference metadata for at least one other object. Snowflake tracks object dependencies in the OBJECT_DEPENDENCIES Account Usage view.

Object dependencies can help customers answer questions about the personal and sensitive data stored in their Snowflake account database objects, such as:

- Where did the data come from? Where is the data located?
- What does the data produce? What does the data consume?
- What associations are there between data elements? What data is impacted by other objects?
- What data sources should we use to develop new customer experience initiatives?
- What data in my enterprise needs to be brought into compliance with industry regulations?
- Where do we have exposure risks that need to be mitigated?
- Who or what business process changed the data? What tool or technology made the change?
- How can data scientists improve confidence in the data that they need for advanced analytics?

Figure 22 shows an anonymized view that was built on the real table named `AAD_SIGNINS` that has sensitive information, such as user emails, addresses, and so on. The anonymized view is generated using the Snowflake [data anonymization](#) capability that is based on the k-anonymity algorithm.

The object dependencies view shows the lineage of the data objects and their dependencies, which helps customers in their data privacy impact assessment.



The screenshot displays the Snowflake Object Dependencies view. At the top, a SQL query is shown in a code editor, selecting columns from the `snowflake.account_usage.object_dependencies` table where the referenced object name is `AAD_SIGNINS`. Below the query, the results are shown in a table with 8 columns: `REFERENCED_DATABASE`, `REFERENCED_SCHEMA`, `REFERENCED_OBJECT_NAME`, `REFERENCED_OBJECT_DOMAIN`, `REFERENCING_DATABASE`, `REFERENCING_SCHEMA`, `REFERENCING_OBJECT_NAME`, and `REFERENCING_OBJECT_DOMAIN`. The results table contains one row showing the dependency of the `K_ANONYMIZATION_AAD_SIGNINS` view on the `AAD_SIGNINS` table.

```
2 select referenced_database,
3        referenced_schema,
4        referenced_object_name,
5        referenced_object_domain,
6        referencing_database,
7        referencing_schema,
8        referencing_object_name,
9        referencing_object_domain
10
11 from snowflake.account_usage.object_dependencies
12 where referenced_object_name LIKE 'AAD_SIGNINS'
13 ;
```

REFERENCED_DATABASE	REFERENCED_SCHEMA	REFERENCED_OBJECT_NAME	REFERENCED_OBJECT_DOMAIN	REFERENCING_DATABASE	REFERENCING_SCHEMA	REFERENCING_OBJECT_NAME	REFERENCING_OBJECT_DOMAIN
GEOFENCING	PUBLIC	AAD_SIGNINS	TABLE	GEOFENCING	PUBLIC	K_ANONYMIZATION_AAD_SIGNINS	VIEW

Figure 22. Snowflake Object Dependencies

Access_History

The [ACCESS_HISTORY view](#) provides the application provider with the following capabilities. (See figure 23.)

- Discovering unused data to determine whether to archive or delete the data
- Validating data changes to notify users prior to dropping or altering a given table or view
- Auditing data access to help comply with regulatory requirements and data governance initiatives

Access History helps customers answer questions about personal and sensitive data stored in their Snowflake account database objects. Following are some examples:

- For read access, such as select queries:
 - What tables are queried the most?
 - When & who last queried a table?
 - What table's users accessed the table the most?
 - What tables are commonly used together?
 - What columns are used the most?
 - What columns are not used at all?
 - What tables have the overall slowest queries?
 - Who saw results from what column? What table? And when and where?
- For write access, such as data manipulation of deletions:
 - What tables act as origins for other tables?
 - What are the most common columns used in the WHERE clause?
 - What are the most common columns used in a JOIN clause for a table?
 - Are these predicates in line with the table clustering keys?
 - When were tables created? When were tables altered? How were tables altered?

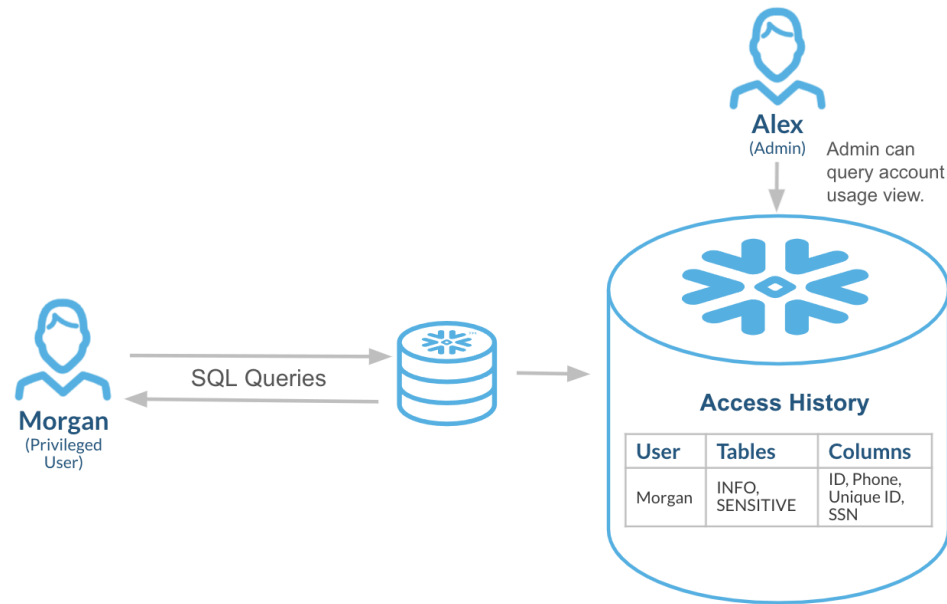


Figure 23. Access History view

Sensitive columns and unauthorized access monitoring

Customers needing additional verification to monitor any unauthorized access to their data should see [Protecting specific columns with Snowflake column-level security](#) or [Hold Your Own Key \(HYOK\) in the customer environment](#). Afterward, the customer should configure their monitoring software, such as their [SIEM](#) in their security operation center, to monitor their HSM API calls, API management/API gateway, policy enforcement point (PEP), and Snowflake account_usage data, like query_history and login_history. Next, customers should set up alerts that fire when there are discrepancies between the query time, the user executing the query, or events from third-party systems, such as an HSM, the API gateway/management, and/or the PEP.

✓	01a436ca-...	select C_CUSTKEY, hex_decode_string(to_varchar(decrypt_raw(C_NAME:ciphertext::BINARY, TO_BINARY(S...	MWHITBY	LOAD_WH	1	X-Small	1053340730...	10:46:00 AM	10:46:05 AM	4.4s	5.5KB
✓	01a436c6-...	select C_CUSTKEY, hex_decode_string(to_varchar(decrypt_raw(C_NAME:ciphertext::BINARY, TO_BINARY(S...	MWHITBY	LOAD_WH	1	X-Small	1053340730...	10:42:27 AM	10:42:38 AM	11.4s	5.5KB

>	5/12/2022	9:46:01.349 AM	1b24bf31d49f6555	get-key-by-name	True	200	3,789.589
>	5/12/2022	9:42:31.660 AM	a43231e4a1ea4df9	get-key-by-name	True	200	6,820.256

Figure 24. Shows the event correlation between Snowflake query_history and the Azure function used to fetch the encryption materials at run time

With that, the customer can trigger an incident process to notify the related parties and stop or prevent such access as shown in the [Sensitive Data Incident Notification and Response](#) section.

Sensitive Data Ingestion Pipeline and Privacy Impact Assessment

This section discusses the guidelines and measures that customers can adopt as part of their "sensitive and PII data ingestion process" to satisfy their data processing security requirements. As mentioned in the [Data governance capabilities](#) section, customers are advised to start by classifying and tagging the sensitive data before they apply the relevant measures.

Note: This section helps customers with data privacy impact assessments better understand their data and their regulatory and compliance requirements.

This is, first and the foremost, an important step to adjusting the additional data protection controls mentioned in the [Security of Data Processing Measures](#) section. In general, applying the maximum security for all types of data is overkill due to the additional administrative overhead and the performance and cost impacts associated with it. Therefore, by classifying and tagging the sensitive data, customers can take the required measures without sacrificing the business value of the data.

With Snowflake, customers can choose to classify, tag, and apply additional measures inside the platform after the data lands in Snowflake, or before the data is ingested in Snowflake, as shown in figure 25.

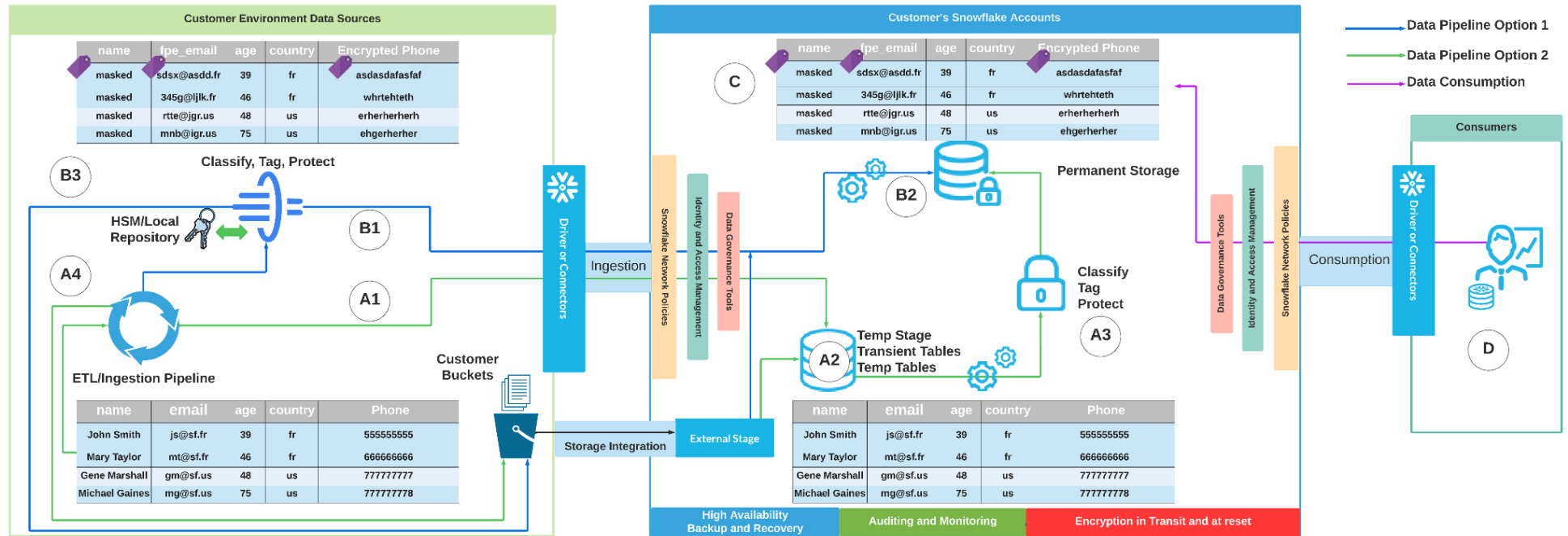


Figure 25. Sensitive data ingestion pipeline

The sensitive/PII data journey starts in the customer environment where the data is generated. The customer has two options:

- A1 (Classification and Protection of Sensitive Data):** (green arrows in the diagram above) Where the customer run the ETL process then ingest the data into Snowflake:
 - A2:** The customer can stage the data in Snowflake where the data will be processed temporarily as per the customer's needs (as discussed in [Snowflake Staging Options](#) section) before being classified, tagged, and protected, as detailed in the [Encryption and Anonymization](#) section, and then stored in the Snowflake [permanent tables](#) (C).
 - A3:** This is where the data is read from the temporary staging area to perform the following actions:
 - Classification:** The customer can use the Snowflake [classification](#) feature. If Snowflake classification does not satisfy the customer's data classification requirements, the customer has two more options:

- Build their own classification functions by using [Snowflake Java UDF](#) or [Python UDF](#) functionality, which allows the customer to classify any data format and localization that is currently not supported by the Snowflake out-of-the box classification.
- Use a third-party tool from the Snowflake [partner ecosystem](#), or any tool that can connect to Snowflake, to classify the data as discussed in **B1**.
- **Tagging:** The classification feature assists with tagging. Customers can update and, additionally, tag certain objects using Snowflake object tagging according to their needs.
- **Protect:** In this step the customer has already identified the sensitive data that needs extra protection by either one, or a combination, of the following as discussed in [Security of Data Processing Measures](#) section:
 - [Role-Based Access Control](#)
 - [Row Access Policies](#)
 - [Tag-based masking policies](#)
 - [Column-level security](#): By column-level encryption, column-level masking, column-level tokenization, and so on. Currently, Snowflake has [built-in functions](#) to support various column-level encryption and hashing functions based on AES, MD5, and SHA algorithms. To support additional encryption algorithms, such as asymmetric encryption, format preserving encryption (FBE), such as fbe_email in the diagram above, customers can leverage [Snowflake Java UDF](#) or [Python UDF](#) functionality, or external functions, to implement the algorithm that satisfies their needs.

Note: Data is always encrypted in Snowflake. The above measures provide additional protection that may be deemed necessary due to customers' GDPR requirements.

- **B1 (Customer Externally Classifies and Protects The Sensitive Data):** (Blue arrows in figure 25) Where the customer runs the ETL process then ingests the data into Snowflake. As part of the ETL process, the customer uses an external tool to classify, tag, and protect the data before it lands in Snowflake. This way, sensitive columns will land classified, tagged, and protected in the customer's Snowflake account.

To decrypt the customer's sensitive data during query time, customers have the following options:

- **Option 1:** Using Snowflake external functions to either decrypt the data outside of Snowflake, or to acquire the encrypted materials to decrypt the data locally in the memory of the virtual warehouses as

shown in the [Snowflake column-level security and Customer Data Key](#) section. With this option, the customer's Snowflake account can use the keys/tokens to decrypt/re-identify at query time. If this is not acceptable, the customers can use option 2.

- **Option 2:** The customer uses policy enforcement points in their environment as discussed in [Split Processing or Multi-Party Processing](#). With this option, the customer's Snowflake account will never be able to decrypt/re-identify the sensitive columns due to the fact that Snowflake has no access to the encryption or token materials.
- **D:** (Purple arrows in figure 25) The consumers can access the data in protected and unprotected format based on the access policies that are defined by the data domain owners, security officers, and/or privacy officers.
- **B3 and A4:** Give the customer the option to ingest data in Snowflake via their preferred cloud or S3 compatible storage. Once the data is in the customer bucket, the customer can use manual or automatic ([Snowpipe](#)) [data load](#) in Snowflake as shown in A1-A3 and B1-B2.

Snowflake staging options

The following table shows the storage options that are available in Snowflake and that helps the customers to satisfy their [Right to Erasure \(Forgotten, Deletion\)](#) and [Security of Data Processing Measures](#) needs.

Table. Data Staging Options

Considerations	Permanent Tables	Transient Tables	Temporary Tables	Internal Stage
Time Travel	0-90 days (configurable)	0-1 days only (configurable)	0-1 days only (configurable)	0 (not configurable)
Fail-safe	7 (non configurable)	0 (non configurable)	0 (non configurable)	0 (non configurable)
Persistence	Until dropped, then it is moved to Time Travel, then Fail-safe	Until dropped, then it is moved to Time Travel for one day only	Remainder of the session	Until stage is dropped or files in the stage are dropped

Summary of sensitive data ingestion pipeline

Table. Sensitive data ingestion options comparisons

Considerations	Inside Snowflake	Outside of Snowflake
Classifications	Snowflake supports PII classification. Snowflake needs to read/decrypt the data to classify it. If columns are encrypted or tokenized outside Snowflake and the customer Snowflake account can't decrypt the data/columns, then the classification will not be accurate.	Customers can classify data in their environment. Customers can classify any data type their classification tool supports.
Tagging	Customers can tag columns, tables, schemas, and so on without the need to read/decrypt the data in the protected columns.	Customers can centrally manage all their data tags and dictionaries across their Snowflake accounts and other data systems in their environment.
Column-level Decryption	Snowflake uses the column encryption keys to decrypt based on the column-level security policy	Snowflake does not use or access any column-level encryption keys.
Column-level Tokenization	Snowflake uses external functions to detokenize the data. Snowflake will temporarily stage the detokenized data in the Internal Stage before returning the results to the customers.	If detokenization is happening in-line as discussed in Hold Your Own Key (HYOK) in the customer environment , then Snowflake will not stage or see the detokenized columns at any time
Anonymization	Snowflake needs to classify the data or manually tag the columns to be able to generate the anonymized views.	Snowflake has no access to the original data. Only the customer is able to re-identify/reconstruct the data using their local repositories .

External Functions	<p>Used in Snowflake to acquire the column-level encryption keys or to detokenize the data. The data is re-identified/reconstructed temporarily in the Snowflake stage before Snowflake returns the results to the customer.</p> <p>Customers can monitor all unauthorized access to those sensitive columns as detailed in the Sensitive Columns and Unauthorized access Monitoring section.</p>	<p>The staged results have only the protected data. Only the customer is able to re-identify/reconstruct the data using their local repositories/HSMs.</p>
--------------------	---	--

Sensitive Data Internal Tagging, Monitoring, and Response

Customer data is the responsibility of the customer. Because Snowflake is largely data blind (or has limited visibility into the content of customer data), the customer is responsible for adopting and implementing the guidelines discussed in this section. Customers should monitor various aspects of their Snowflake accounts, along with other enterprise applications as shown in [Regularity testing and evaluating the Security of Customer Data](#).

Note: Read the [Sensitive Data Ingestion Pipeline and Privacy Impact Assessment](#) section and apply the measures discussed before working through this section.

Note: This section covers incident notification regarding the customer's stored sensitive data. It does not address incident notification for the Snowflake platform. For information about Snowflake incident detection, visit [Snowflake security addendum section 7](#).

Setting up alerts and notifications

Customers can set up auditing and monitoring rules, such as:

- Queries to monitor read-and-write operations on sensitive data objects by using [object dependencies](#), [tag references](#), and [access history](#).
- Queries to monitor and audit tag-based policy changes and what columns/tables are impacted.
- Queries to monitor the sensitive data flow inside Snowflake using [object dependencies](#).
- Queries for monitoring, as discussed in the [Login History](#) and [Query History](#) sections.
- Auditing and monitoring the use of external functions to detokenize or to programmatically bring the encryption keys that are used to decrypt the protected columns as discussed in [Sensitive columns and unauthorized access monitoring](#) section.

Additional auditing and monitoring rules are discussed in [Regularity testing and evaluating the security of customer data](#) section.

Note: The above is not an exhaustive list. Customers may build additional queries to satisfy their needs. Customers should work with their Snowflake contact to get the latest version of the Snowflake *Security Features Checklist*, which has a set of auditing and monitoring queries that can be used as a starter kit.

After building detection queries, the next step is to use them for alerting to trigger the appropriate response within the customer's [security operation center](#) in case of suspicious activities, such as:

- Entities are accessing Snowflake from botnet IP addresses by leveraging Snowflake Data Marketplace botnet datasets, such as [Red Sky Alliance datasets](#)
- Entities are trying to access unauthorized sensitive data objects and failing (failing queries in Query History)
- Entities are trying to copy data out of a sensitive data object
- Utilizing Snowflake Data Marketplace security datasets, such as [KELA's DARKBEAST](#), [Darkfeed by Cybersixgill](#), and [others in the security dataset category](#)

Customers can leverage the Snowflake [native alerting feature](#) that can:

- Send a notification email to the customer's security operation center to trigger an [incident response procedure](#)
- Execute a query to revoke access to the users violating the access policies
- Execute a query to drop rows or modify the impacted data

The above is just a sampling of the type of detection queries and alerting that customers can set up on sensitive data objects and on the top of the customer's [account usage views](#).

Sensitive Data Incident Response (IR) Process

To be able to build a sensitive data incident response process, customers need to build the proper detection queries in their Snowflake account(s) as discussed in the [Customer data incident alerting and notification](#) section. In addition, the customer must integrate such detection and alerting with their security operation center to have a full view of an incident's scope.

Figure 26 shows a template for a sensitive data incident response process. When reviewing this template, consider that:

- The process must be part of a comprehensive customer incident response process and must not be used separately.
- The process is provided as an example and is not a complete IR solution.
- The customer must select the appropriate items for each step according to their compliance and policy requirements.
- The process is about sensitive customer data stored in Snowflake, and is separate from the notifications about security incidents that Snowflake sends for the Snowflake Service. (To learn about the Snowflake security incident response notification commitments, see section seven in the [Snowflake security addendum](#).)

Snowflake Technical Tools for Protecting Sensitive Customer Data

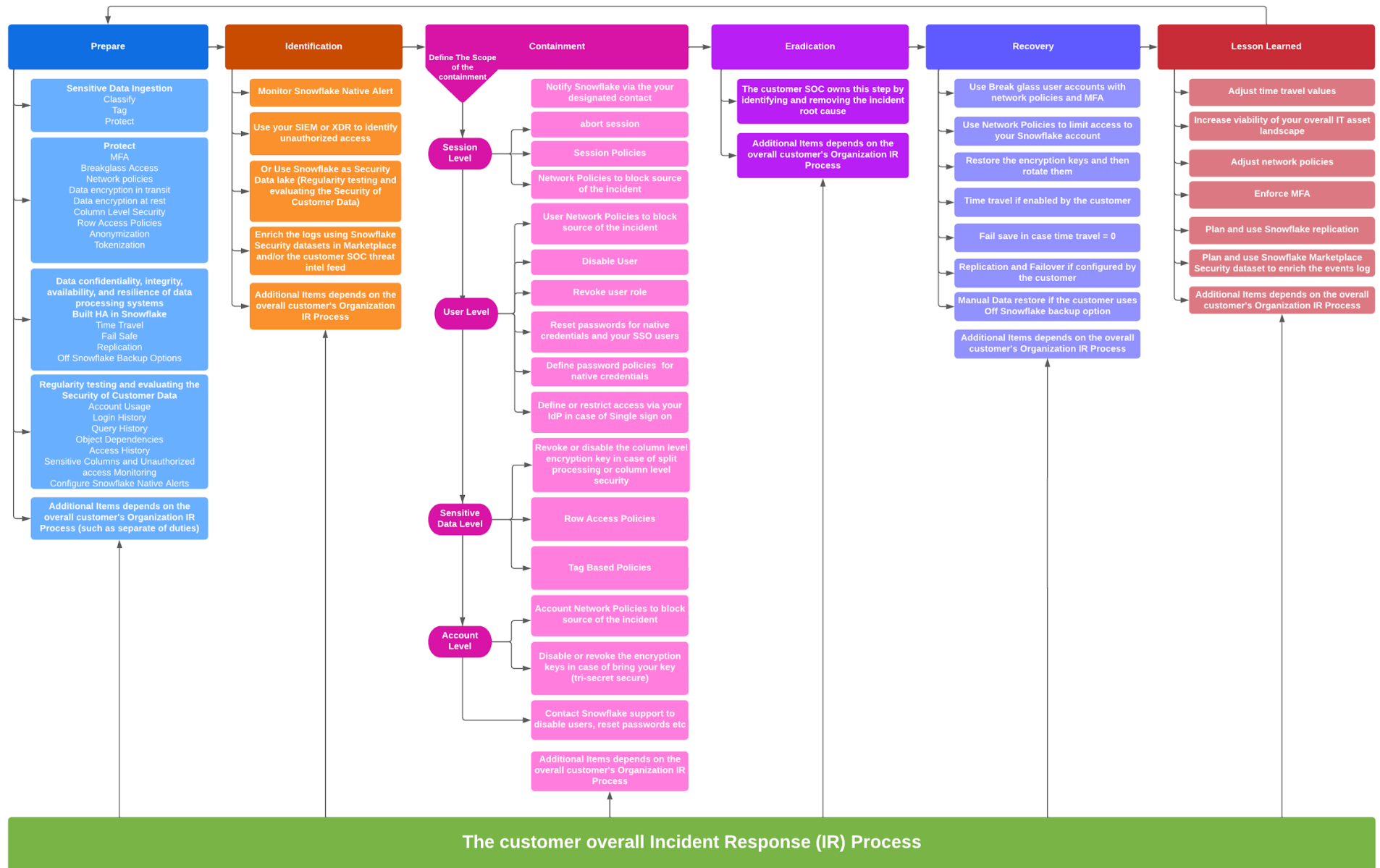


Figure 26. A sensitive data incident response process

The following points describe the IR process example in figure 26, which is based on the [NIST](#) Computer Security Incident Handling Guide:

- **Prepare phase:** Relies on the guidelines in the [Summary of sensitive data ingestion pipeline](#) and [Security of Data Processing Measures](#) sections—for example, the break glass access process that utilizes user accounts with double MFA and network policies to restrict access from specific trusted locations only. All usage of such user accounts should be closely approved, monitored, and audited.
- **Identification phase:** Relies on the guidelines in the [Regularity testing and evaluating the Security of Customer Data](#) and [Sensitive Data's Incident Notification and Response](#) sections.
- **Containment phase:** The customer notifies Snowflake through any means such as support, their Snowflake point of contact, etc. The IR process should be able to identify the incident impact radius and then take the appropriate actions based on the scope of the impact as shown in figure 26:
 - **Session-level containment:** Actions that impact the current established Snowflake sessions that are identified as the source of the incident.
 - **User-level containment:** Actions that impact the users that are identified as the source of the incident.
 - **Sensitive data-level containment:** Actions that impact the identified sensitive data objects that are affected by the incident.
 - **Account-level containment:** Actions that impact one or more Snowflake accounts affected by the incident. Action at this level stops all operations on the identified Snowflake accounts, for example: running queries will fail, and the account will not be usable during the containment period. Access and operation can be restored later, however, based on the recovery procedures.
- **Eradication phase:** Once the root cause has been identified, the customer uses the overall organization or process to eradicate the source of the incident and start the recovery phase.
- **Recovery phase:** Once the eradication phase mitigates the source of the incident, the recovery phase starts. This process depends on the scope of the incident and what action was taken in the containment phase. For instance, the break glass access may be used:
 - To update network policies
 - Restore and then rotate encryption keys

In case of data corruption, use Time Travel, Fail-safe, and/or replication to recover the data as discussed in the [Customer data processing confidentiality and integrity](#) section.

- **Lesson learned phase:** The customer considers and applies additional controls that may be preventive going forward, such as:
 - Adjusting Time Travel
 - Enforcing MFA
 - Using Snowflake replication
 - Considering the [Snowflake Data Marketplace security dataset offerings](#)
 - Adjusting network policies
 - Using private connectivity
 - Considering using [Snowflake for security](#)
- **Integration:** All of the above integrated into the customer's overall organizational IR process. This template should not be used in isolation.

Data Subject Access Right (DSAR)

Snowflake customers have full control over the data that is ingested into Snowflake. This includes any sensitive data objects that have been created. Snowflake customers can satisfy their data subject access rights by following the guidelines highlighted in these sections: [Sensitive Data Ingestion Pipeline and Privacy Impact Assessment](#), [Snowflake column-level security and Customer Data Key](#), and [Split Processing or Multi-Party Processing](#).

For example:

- Customers can classify and categorize sensitive data processing as discussed in the [Sensitive Data Ingestion Pipeline and Privacy Impact Assessment](#) section.
- By following the [Sensitive Data Ingestion Pipeline and Privacy Impact Assessment](#) section guidelines, customers can audit and monitor the purpose of the processing, then use [Restriction of Sensitive Data processing](#) to limit or remove the sensitive data.
- Using Snowflake [access history](#), [tag references](#), and [object dependencies](#), customers can see the flow of the sensitive information in Snowflake from ingestion to deletion.
- Customers can use Snowflake capabilities to implement DSAR to [restrict sensitive data processing](#), [rectify the sensitive data](#), or delete the sensitive information
- Snowflake customers can choose to share sensitive data across borders by leveraging Snowflake [data sharing capabilities](#) and applying various data protection tools on the data provider side, such as masking, anonymization, tokenization, and row access policies as discussed in the [Snowflake column-level security and Customer Data Key](#) section and the [Split Processing or Multi-Party Processing](#) section.

Note: As discussed in the [data encryption in-transit](#) and [data encryption at-rest](#) sections, even without additional protection, customer data is always encrypted in-transit and at-rest. Furthermore, data sharing is a capability that the customer controls. The customer must enable it on the data provider side, and authorize the consumers who are allowed to access the shared data objects.

Right to Erasure (Forgotten, Deletion)

The data subject has the right to petition the data controller (in this case, Snowflake customers) to delete/erase the data subject's personal data. The data controller then must erase the petitioner's personal data.

There are three main tasks for successful "Right to Be Forgotten" or "RTBF" (also known as "deletion" or "right to erasure") implementation:

1. Define what deletion, forgotten, and erasure technically means.
2. Define the technical meaning of "undue delay," for example "*the period of time between when the controller receives the deletion request and when the deletion operation is executed, as defined in (1).*"
3. Follow the guidelines in the [Sensitive Data Ingestion Pipeline and Privacy Impact Assessment](#) section.

The following sections expand on these main tasks and focus on how to delete the data subject's data using various Snowflake tools.

Technically define what it means to be deleted or forgotten

It is the customer's responsibility to define what *deletion* or *forgotten* means in technical terms. The following table lists a few considerations when it comes to Snowflake.

Table. Data deletion options and considerations

Deletion Technical Options	Considerations
Crypto Deletion or Crypto Shredding	The encrypted sensitive columns still exist or are scheduled to be purged from storage at a later stage. The encryption keys or tokens used to decrypt the data, however, are destroyed and no longer retrievable. Customers here can choose either Snowflake column-level security and Customer Data Key or Split Processing or Multi-Party Processing solutions.
Data Masking At Run Time	The sensitive data still exists or is scheduled to be purged from storage at a later stage. The sensitive data, however, is masked at query time and cannot be seen by unauthorized parties.

Block User Access	Customers can implement RBAC with row access policies to block access to specific records while being scheduled for deletion. (See the diagram below.)
Deletion from Snowflake Data Storage	Customers should tune their Time Travel value according to their business continuity plans and according to their compliance and regulation requirements. The customer can only access deleted data or previous versions of the data without Snowflake tech support during the Time Travel period. Once the Time Travel time is passed, the data is moved to Fail-safe and stays there for a seven day (non configurable) period. During those seven days, however, the customer cannot directly access Fail-safe data storage. To access data in Fail-safe, only an authorized customer representative can open a support ticket. Snowflake support will help the customer restore access to the encrypted micro-partitions. Once Time Travel and Fail-safe time has passed, sensitive data is no longer available or retrievable from Snowflake storage.

Note: The above "Data deletion options and considerations" table is not an exhaustive list of technical options.

Define the deletion/forgotten time frame

Customers must define the time frame for the deletion of sensitive data, which is when the data is no longer accessible after submitting the deletion/forget request.

Right to be forgotten guidance (RTBF)

To find the appropriate right-to-be-forgotten solution, as shown in figure 27, see the following sections: [Sensitive Data Ingestion Pipeline and Privacy Impact Assessment](#), [Technically Define Deletion-Forgotten](#), and [Define the Deletion-Forgotten Time Frame](#).

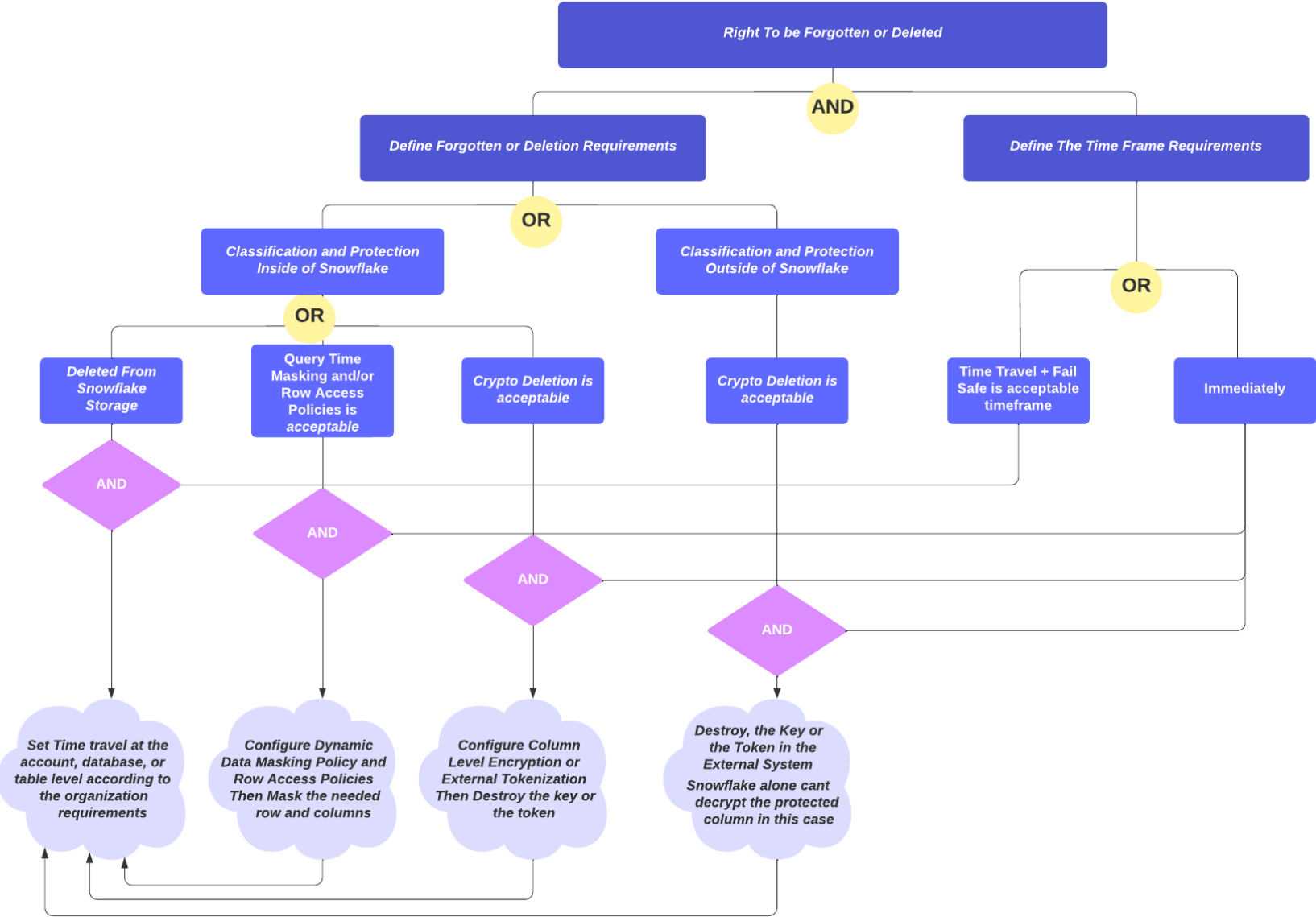


Figure 27: RTBF technical measures

Restricting sensitive data processing

As discussed in the [Sensitive Data Ingestion Pipeline and Privacy Impact Assessment](#), Snowflake provides customers with a wide set of tools to build their data models. These tools enable them to:

- Classify the sensitive data
- Tag the sensitive data
- Protect sensitive data by restricting data processing by:
 - Masking and externally protecting sensitive columns as discussed in the [Protecting specific columns with Snowflake column-level security](#) section
 - Masking sensitive cells ([Conditional masking](#))
 - Updating or deleting sensitive information using [SQL](#), [Java](#), or [Python](#)
- Audit and monitor sensitive data flows within Snowflake, and provide their end customers with evidence of restricted data processing to only what is needed and based on their consent as discussed in the [Regularity testing and evaluating the security of customer data](#) section.

Customers can build their processes around restricted sensitive data processing and use Snowflake tools as mentioned above to implement such processes.

Data Rectifications

Snowflake customers can use [SQL](#), [Java](#), or [Python](#) to update any type of data stored in their Snowflake accounts. Therefore they can update sensitive data as per compliance and policy requirements based on data subject requests.

Note: Customers should consider the steps discussed in the [Sensitive Data Ingestion Pipeline and Privacy Impact Assessment](#) section.

Notification of Rectification or Erasure of Sensitive Data or Restriction of processing

Snowflake customers have all the required tools and audit logs needed to set up notification processes around [Restricting sensitive data processing](#), [Data Rectifications](#), and [Right to Erasure \(Forgotten, Deletion\)](#).

Customers can use the tools discussed in the [Sensitive Data Ingestion Pipeline and Privacy Impact Assessment](#) section and the [Sensitive Data Incident Notification and Response](#) section to build queries to answer questions, such as:

- What are the sensitive data flows in Snowflake using access history and object dependencies?
- When and how is the sensitive data updated inside the data objects? Who is responsible?
- How is sensitive data processing minimized? For instance, if a sensitive column is not being used by the business lines, it should be removed from the data objects.
- When was sensitive data deleted? Who deleted it? How was it deleted? Normal delete or update? Crypto deletion? Masking? These and other questions are based on the options discussed in the [Right to be forgotten guidance \(RTBF\)](#) section.

Note: The questions listed are intended to show the kinds of queries that customers can build. This is not a comprehensive list of questions that customers should consider.

Customers can build notification processes using one or a combination of the following:

- [Snowflake native alerting](#).
- Using the customer data application built on the top of Snowflake. For more information, please refer to the [Snowflake Security and Identity & Access Management for Data Applications](#) whitepaper.
- Using customer compliance, auditing and/or monitoring tools as discussed in the [Regularity testing and evaluating the security of customer data](#) section.

Data Portability

Snowflake customers can load and unload [commonly used and machine-readable data formats](#) in their Snowflake accounts, such as:

- Structured (CSV, TSV, *etc.*)
- Semi-structured (JSON, Parquet, Avro, ORC, *etc.*)
- Unstructured data (images, video, and audio, *etc.*)

Data portability in Snowflake can be achieved two ways:

1. Snowflake is built on top of the three major cloud providers (AWS, GCP, Azure). Therefore, customers have the freedom to move their sensitive data across cloud providers by leveraging cross-cloud [data unloading](#), [drivers and connectors](#), and/or [replication](#).
2. Snowflake customers who wish to terminate their Snowflake services can use the [unload data](#) capability or Snowflake [drivers and connectors](#) to export their sensitive data outside of Snowflake using the above [standard format](#) as discussed in the [Off Snowflake backup options](#) section.