A Security Engineer's Guide to Data Masking

Overview

Security engineers are tasked with securing sensitive information across diverse engineering, production and analytical environments. In the modern data infrastructure landscape, this is data stemming from databases, data warehouses, and data lakes, which traverses through various applications, ETL jobs, BI tools, notebooks, and more. This often spans multiple teams, such as data, SRE, and DevOps, responsible for data maintenance and management.

This white paper can serve as a guide for security engineers that are planning to implement data masking as a way to secure their sensitive data.

- Different masking strategies and how it compares to other data devaluation techniques
- A practical framework for implementing data masking
- Usability and management considerations for data masking at scale

Securing sensitive information is a paramount concern for security engineers, and this white paper enables security engineers to mitigate risks associated with data sprawl without impeding operational functionality, and provides a framework they can use for collaborating with their peers and stakeholders.

Data Devaluation and Data Masking

Data devaluation plays a pivotal role in fortifying the security of sensitive information within complex operational environments. It involves transforming original data into a modified format that retains usability for authorized purposes while simultaneously reducing the sensitivity of the data.

Drivers for Data Devaluation

- **Privacy and Compliance:** Adhere to stringent data privacy and compliance laws such as GDPR, CCPA, HIPAA etc. Devaluation ensures compliance with regulatory requirements by safeguarding personally identifiable information (PII) and other sensitive data.
- **Risk Mitigation:** Reduce the potential impact of a security breach by rendering sensitive information less valuable to unauthorized access, safeguarding against unauthorized use or exposure of critical data.
- **Trust and Reputation:** Devaluation measures assure customers their data is handled responsibly and securely, fostering trust and confidence in the organization.
- **Safe Data Sharing:** Allow organizations to share data more broadly, both internally and externally, without compromising its sensitivity. This facilitates collaboration, analysis, and reporting while protecting the confidentiality of information.
- **Reducing Data Handling Costs:** Reduce the costs associated with storing and securing vast volumes of sensitive information, optimizing resources and infrastructure.

Typical Data Devaluation Techniques

Below we list the most common techniques for data devaluation, and the relative level of complexity in maintenance and implementation.

1 Data Masking

- **Description:** Data masking involves the modification of data to conceal sensitive information while maintaining its usability for authorized purposes.
- **Complexity:** Moderately complex to set up and with light ongoing maintenance to ensure consistent application across various systems and databases. Regular updates and validation are necessary to maintain data integrity.

2 Encryption

- **Description:** Encryption encodes data to make it unreadable without the corresponding decryption key, ensuring secure transmission and storage.
- **Complexity:** Initial implementation can be complex, but maintaining encrypted data is relatively straightforward. Key management can be complex and mistakes here can result in data loss. Additionally, in some cases, changes to encryption keys can require the decryption and re-encryption of the entire dataset.

3 Tokenization

- **Description:** Tokenization replaces sensitive data with unique tokens, preserving referential integrity while protecting sensitive details.
- **Complexity:** Moderate-to-high in complexity for initial set up, particularly in ensuring consistency across systems. Maintenance involves managing tokenization mappings and ensuring synchronization between tokenized and original data.

4 Anonymization

- **Description:** Anonymization removes identifying information, making it impossible to link data to an individual or entity.
- **Complexity:** Moderately complex in the initial setup, with ongoing efforts required to ensure data remains anonymized and consistent across different data sources. Regular checks and updates are necessary to maintain the anonymization process.

5 Psuedonymization

- **Description:** Pseudonymization substitutes identifiable information with artificial identifiers, maintaining data usability for authorized purposes.
- **Complexity:** Quite complex in setup, particularly in ensuring consistent usage of pseudonyms across systems. Regular monitoring and maintenance are required to maintain the linkage between pseudonyms and original data.

The choice of technique may depend on the specific data environment, regulatory compliance needs, and the level of sensitivity attached to the information. Organizations must balance the effectiveness of these devaluation techniques with the effort required for their implementation and ongoing maintenance to achieve a robust data protection strategy.

Often data devaluation and data obfuscation are used interchangeably. There is a subtle but important practical difference between the two:

	Data Devaluation	Data Obfuscation
Goal	Safeguard sensitive data against breaches, while preserving data usability	Conceal specific information from everyone, without altering its overall structure
Techniques	Masking, encryption, tokenization, anonymization, pseudonymization	Masking, scrambling, randomization, shuffling

While both approaches serve the purpose of protecting data, Data Devaluation primarily aims to reduce the value of sensitive information to unauthorized users while maintaining usability, whereas Data Obfuscation focuses on making data harder to interpret, whether sensitive or not.

Why Data Masking is Popular

Data masking has emerged as a highly favored technique for devaluation due to its adaptability, effectiveness in preserving data usability, and its diverse application across various industries and data environments. It aligns with the requirements of numerous data privacy regulations, such as GDPR, HIPAA, and PCI DSS. It allows organizations to comply with these standards by obscuring sensitive details while ensuring data remains useful for legitimate purposes.

One of the unique advantages of masking is its adaptability - data masking can be implemented in a variety of ways that helps provide an easier alternative to other forms of data devaluation and obfuscation. Common data masking techniques include:

- **Static Masking:** Replaces original data with consistent, static values, such as replacing all occurrences of a specific data type with the same masked value.
- **Dynamic Masking:** Involves generating masked data in real-time when it is queried, potentially even producing different masked outputs for each query or user, leaving the original data as-is.
- **Format-Preserving Masking:** Modifies data while retaining its format, ensuring the length, structure, and data type of the masked information remain unchanged (could be dynamic or static).
- **Shuffling:** Rearranges the order of data records, making it more challenging to identify specific information while preserving the dataset's structure.
- **Character Masking:** Alters specific characters within the data, such as replacing specific digits in credit card numbers or phone numbers, while retaining the data's general format.

These techniques offer a choice between security and utility, making masking a popular choice for safeguarding sensitive information.

Masking for Data Infrastructure

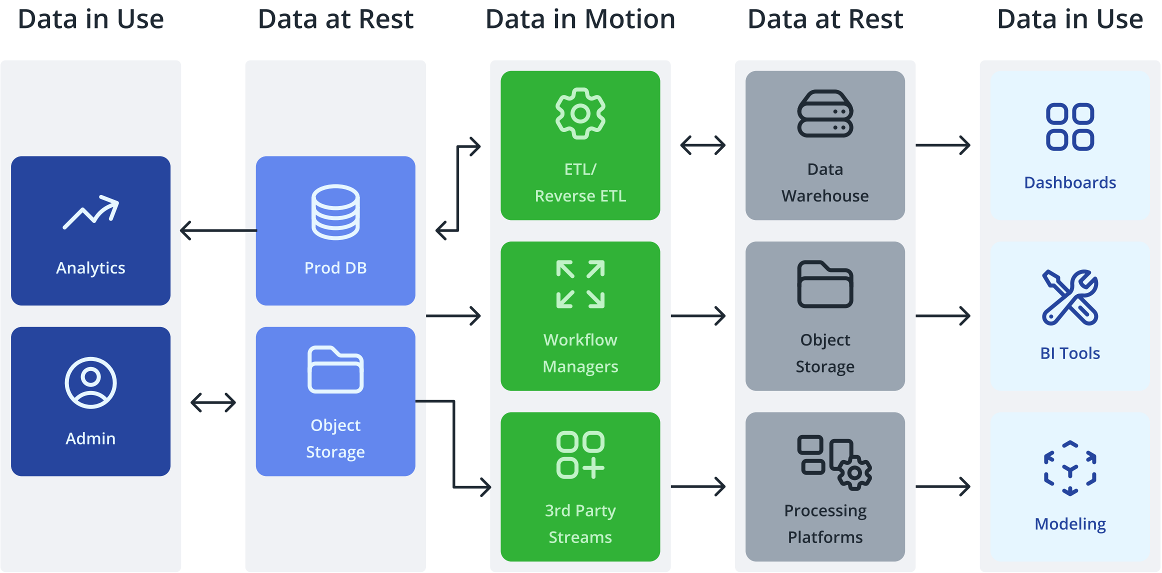
Implementing masking in a typical data infrastructure is a complex undertaking. The modern data stack is an intricate ecosystem characterized by an array of rapidly evolving components and interdependencies. It comprises diverse databases, cloud services, streaming platforms, containers, microservices, and more, all while being in a constant state of flux. The ever-changing nature of these interconnected elements, coupled with frequent updates, patches, and new integrations, contributes to a highly complex and agile system where components are consistently evolving, challenging the stability and security of the overall infrastructure. Implementing data masking in such a highly dynamic infrastructure is challenging, and often security teams understandably don't know where to even begin.

A Framework for Where to Implement Masking

Despite the sophistication of use cases the various data tools enable, one can break these down into three broad categories security teams are well versed with:

- **Data at Rest:** These are database, and database-like repositories (eg S3) where data is stored in some format and used for some purpose. Collectively, these can be termed as data repositories.
- **Data in Motion:** These are pipelines and various connectors (eg Airflow) that enable different kinds of automated processing on the data and move them from one data repository to another. Collectively, these can be termed as transformation tools.
- **Data in use:** These are typically clients which enable different people in the organization to visualize and experiment with the data. The consumers of the data itself can be categorized into the following three groups:
    - **Privileged Users:** These are employees with administrative privileges over the various data repositories.
    - **Business Users:** These are employees who access the data through some visualization tool, which generally connects to a repository via a service account.
    - **External Users:** These exist outside the organization, and access the data through an application, almost always through a service account.

If we trace the data flow, we can come up with the following representation of the data stack:



The goal of any masking strategy is to manage, at scale, what type of masking policy to apply depending on the data and the consumers.

- Data is masked both in transit and in use.
- The same data can be accessed by different users, with different masking strategies applied.
- No data is accessible without ensuring proper policy checks.

Implementing Data Masking Policies

Masking policies in databases can be effectively implemented through two primary approaches, each offering distinct advantages in managing sensitive data within the system.

1. **Database UDFs and Stored Procedures:** Employing User Defined Functions (UDFs) and Stored Procedures within the database allows for an internal application of masking policies. These functions and procedures execute within the database itself, offering a direct and controlled method to apply data masking. By integrating with the Role-Based Access Control (RBAC) model, masking functions are linked with not just the underlying dataset but also the database roles which are associated with specific access privileges.

This approach ensures that only authorized users or roles have access to the unmasked data while enabling seamless application of masking rules to relevant data elements. It provides a granular level of control and facilitates the application of consistent and standardized data masking policies within the database environment.

2. **External Authorization Service:** Alternatively, organizations may opt for an external authorization service to implement and manage masking policies. This approach allows for the centralized management of policies across multiple databases or systems. The external service applies policies outside the database, often decoupled from specific database roles, providing a more centralized and independent control over data access and masking rules. By separating the enforcement of masking policies from the database itself, this method offers flexibility in managing policies across diverse databases and environments. It also allows for a uniform application of policies irrespective of the underlying database structures or technologies used, offering a more scalable approach for enterprises managing multiple databases or systems.

These approaches cater to different needs and environments. Internal application of masking policies within databases allows policies to be applied on all access paths, but the policies are highly custom to the specific database and how UDFs are managed. On the other hand, employing an external authorization service provides a centralized and scalable solution, enabling consistent policy application across multiple databases and systems while maintaining a degree of independence from individual database functionalities, but depending on the vendor it may or may not support all access channels.

Masking for Data Infrastructure

Any masking strategy has tradeoffs. However, there are common considerations that every team should be aware of:

1. **Policy Granularity:** Most masking strategies operate at the column level, applying rules to specific data elements. While rare use cases may require row-level masking, the focus on column-based approaches often meets the majority of needs.

Granularity considerations should align with the sensitivity of the data and the intended use to avoid overcomplicating the masking process.

2. **Management challenges due to database roles:** Policies are invariably linked to database roles, introducing complexities in management. Database roles can be cumbersome to administer and maintain. This could lead to challenges in assigning and revoking role-based access, potentially complicating the enforcement of masking policies.
3. **Policy replication for applications:** For effective implementation, policies need replication within applications, which map various distinct users to a singular service account. This approach facilitates consistent data access while ensuring the application of masking policies across multiple user profiles. However, this replication adds an additional layer of complexity in managing and synchronizing policies between databases and applications.
4. **Policy Ownership:** Writing and managing complex masking policies demands specialized expertise. Often, only data teams possess the required skills, posing challenges in ownership and management. Ensuring a clear understanding of roles and responsibilities between data and application teams becomes crucial for successful policy maintenance.
5. **Data Labeling:** Successful implementation of data masking strategies relies on a robust data labeling strategy. Data must be accurately labeled to identify sensitive elements, enabling effective and generalized application of masking policies across databases and applications.

Awareness of these points is crucial for companies intending to implement data masking in databases effectively. Addressing these considerations can significantly impact the success and efficiency of the applied data masking strategies, optimizing both security and operational functionality.

Cyral

About

Cyral provides controls for privacy, compliance, governance and protection thereby reducing risk, complexity, and cost for managing structured data. The Cyral platform discovers data, unifies access controls for users and applications, enables fine-grained authorization policies and provides complete monitoring and reporting. This comprehensive coverage enables risk-based governance, limits the blast radius of data-related incidents and reduces overhead and costs. Cyral's technology allows customers to implement data security controls using their existing, centralized entitlements, thereby simplifying administration and automating remediation. Customers use Cyral to accomplish least privilege, data minimization, spillage prevention and Zero Trust.

For more information, visit **www.cyral.com** or follow **@CyralInc** on X.