Advanced Data Obfuscation with Libelle **Data***Masking*

Architecture, Features, and Benefits

Libelle LLC, Atlanta GA, USA
☎ +1 (770) 435 1101 ✉ sales.americas@libelle.com

Table of Contents

Introduction

This whitepaper outlines how data masking addresses organizations' common data privacy and data protection requirements and describes how Libelle's **Data***Masking* (LDM) software solution meets and supports those requirements.

Why Data Masking?

Market drivers for data masking are not only regulatory requirements such as ITAR, GDPR, or PCI DSS. Data masking is a key component of ensuring maximum protection of sensitive data to reduce the risk of data breaches. It is a common-sense approach to avoid storing sensitive data in places where they are not required thus minimizing risks of data exposure.

What is Data Masking?

Data masking is the process of replacing original data with modified content to remove sensitive data irreversibly from systems. Due to its nature, the masking typically occurs on non-production systems such as Development or Sandbox systems. Also, masking occurs on data analytics scenarios where the masking process retains data value while protecting sensitive data. Data masking is also often referred to as data obfuscation, or data scrambling. Sometimes data hiding is improperly referred to as data masking, but it is an entirely different process. Data masking is not data encryption; once the data is masked, it is permanently changed.

Term	Methodology	Retains orig. Data?	Can be reversed?	Human Readable?
Data Masking	Replace original data with random data while maintaining original structure and readability.	No	No	Yes
Data Hiding	Provide different views on content for different users dynamically via shadow tables	Yes	n/a	Yes
Data Encryption	Convert data into cipher text typically with an encryption key that can be accessed with a key	Yes	Yes	No
Anonymization or Pseudonymization	Replace original data by blanks, asterix, or de-identified identifier. (Subset of Data Masking)	No	No	No

Table 1 – Data Masking Terms

Data masking replaces original data with random characters, while retaining the original patterns so that data is still valid for testing and analytics. Post-masking, names still look like names, although they are completely random. Credit cards still retain valid check digits, although the numbers were randomly created.

Data Masking for Non-Production Systems

Non-production systems or production support systems are typically exact replica of production systems and are used to support the lifecycle of application development, testing, and rolling out features for production. Some application providers design production support systems as exact copies of production systems. Others, such as SAP® ERP, have an elaborate process of building and maintaining production support systems referred to as homogeneous system refresh.

Non-production systems such as Quality Assurance, Sandbox, Development, Training, or Project systems generally have lowered security thresholds. More users have access to these systems, including off-shore development teams, and the user-access level is often broader. Project systems are sometimes given to a project team with root-level access, bypassing security policies set for accessing the same data in production. Many high-profile data breaches originated by unauthorized persons accessing non-production systems. Data masking sanitizes non-production systems by obfuscating production data, dramatically limiting the risk of breach or compromise.

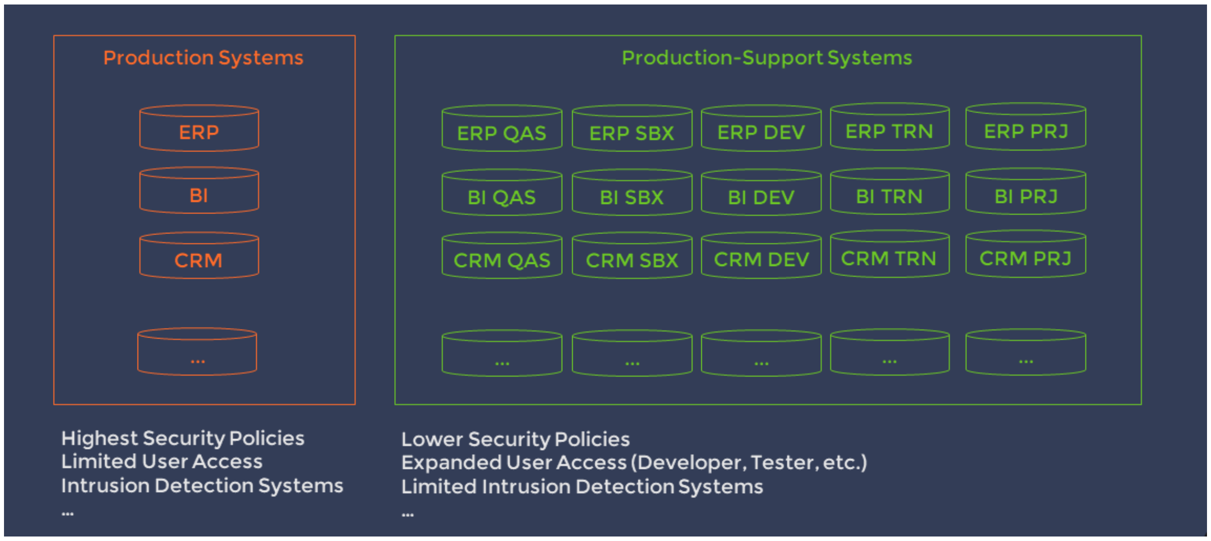


Figure 1 – Production vs. Non-Production System

Data Masking for Analytics

Machine learning and big data provide unprecedented opportunities for data analytics. For example, medical data from millions of patients can be utilized for the treatment and eradication of diseases. However, the question of how personal data is protected and accessed is often unanswered. Data masking is an opportunity to apply intelligent algorithms to obfuscate personal information, while leaving valuable data for analytics intact. For analytic purposes, fields with numeric values (e.g. birth dates) can be masked within certain ranges, while names are completely anonymized by replacing them with random names. Thus, analytics applied to properly masked data leaves a perfectly valid data source from which to derive trends, without the threat of access to any PII or PHI.

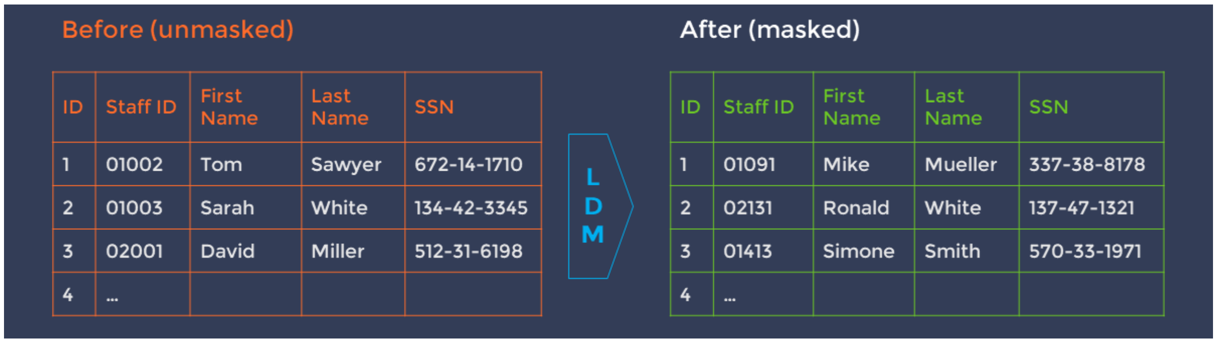
Data masking is executed prior to running analytics on sensitive data. Algorithms are implemented in a way that maintains meaningful data sets which are fully anonymized.

Libelle ***Data**Masking* (LDM) Overview

LDM is a software solution from Libelle to obfuscate data. LDM installs on a masking server in a customer data center; once installed, connections to the databases that require masking are configured. Masking profiles and algorithms are set up once, and masking can be executed on a database level repeatedly on an ad-hoc basis. Key components of LDM are:

- **LDM Master Server:** A software installation installed on Linux, Windows, or UNIX that holds configuration data, templates, connection data. LDM server is usually installed on a customer-controlled environment that allows connection to the database.
- **LDM User Interface:** A web-based console to setup configuration and execute masking. Users authenticate with the master via username/password combinations or via LDAP/ Active Directory.
- **LDM Masking Profiles:** A list of tables, columns, and fields that are included for masking.
- **LDM Masking Algorithms:** Pre-configured algorithms to anonymize data such as replacing numbers, names, etc. A list of algorithms is outlined later in this whitepaper.
- **LDM Reference Database:** A Libelle-provided reference database with names, addresses, and other data that is used to replace original data. Customers can use the Libelle reference database, or their own source for masking.

A simple masking use case is illustrated below; this configuration masks fields for Staff ID, First Name, Last Name, and Social Security Number. LDM masks data consistently, so secondary indices such as Staff ID can be masked identically across the database. In this case, a masked Staff ID may refer back to a masked or unmasked field in other tables or other data stores.


Before (unmasked)
After (masked)
LDM

Before (unmasked)

ID	Staff ID	First Name	Last Name	SSN
1	01002	Tom	Sawyer	672-14-1710
2	01003	Sarah	White	134-42-3345
3	02001	David	Miller	512-31-6198
4	...			

After (masked)

ID	Staff ID	First Name	Last Name	SSN
1	01091	Mike	Mueller	337-38-8178
2	02131	Ronald	White	137-47-1321
3	01413	Simone	Smith	570-33-1971
4	...			

Figure 2 – Data Masking Example

After installation and configuration, masking is executed on an ad-hoc basis after a system refresh, or every time data is loaded to the system. Masking irreversibly replaces values in the database with random data generated by the masking algorithms.

LDM Software Architecture and Features

LDM is based on the powerful Libelle platform that is developed and maintained by Libelle. It is a fully-fledged application platform with a database where all configurations are stored. LDM runs on Linux, Windows, or UNIX. LDM administrators or operators access functionality via a web-frontend (UI), or a command-line interface (CLI). Most everything that is configurable in the UI can be programmatically accessed or executed with a CLI which allows for automating or integrating the masking process. The master server comes with built-in database connectors to natively access most commonly used databases as outlined below. Everything that is configured to be masked in LDM is referred to as data store.

Supported Operating Systems	Supported Databases	Supported Flat Files
Linux®, Windows®, AIX®, Solaris®, HP-UX®	IBM® DB2, Oracle®, SAP ASE®, SAP HANA®, SAP® MaxDB®, Microsoft® SQL Server, MySQL, other database via ODBC	ASCII, CSV, JSON, XML

Table 2 – Supported Environments

All configuration and execution data are stored on the LDM master server. It is typically configured as a single separate stand-alone server which connects to multiple data stores (databases or flat files) that require masking.

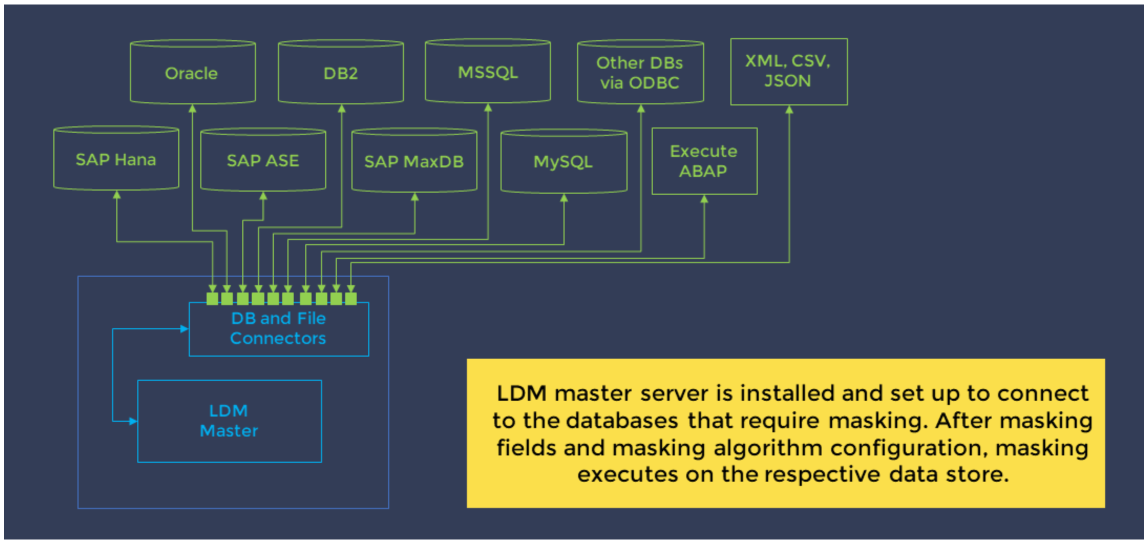

Oracle
DB2
MSSQL
Other DBs via ODBC
XML, CSV, JSON
SAP Hana
SAP ASE
SAP MaxDB
MySQL
Execute ABAP
DB and File Connectors
LDM Master
LDM master server is installed and set up to connect to the databases that require masking. After masking fields and masking algorithm configuration, masking executes on the respective data store.

Figure 3 – Master Server Configuration

LDM master server is installed and setup to connect to the databases that require masking. After masking fields and masking algorithm configuration, masking executes on the respective data store.

LDM Masking Profiles

LDM masking profiles are collections of data in a data store that will be masked. Profiles are a generic term used in LDM to categorize data by certain groups; each masking profile then has certain attributes assigned. For example, the profile "Bank Data" includes attributes such as IBAN number, Bank Account Number, SWIFT code, etc. Each profile has attributes that link characteristics and masking algorithms to tables and fields in a specific masking configuration. There are numerous default profiles available in LDM which are regularly enhanced and expanded, and users can add custom profiles to their LDM installation.

Profile	Attribute
Bank Connection Profile	BIC
	Bank Code
	IBAN Number
	Bank Account Number
	Country Code ISO
	SWIFT Code
Location Information Profile	UTM Coordinates
	WGS Coordinates
	Country
	Place of Birth
	Post Office Box
	Postal Code
	Phone
	Street Address
	City
Date Profile	Year of Birth
	Month of Birth
	Day of Birth
	SAP Date Format YYYYMMDD

Table 3 – Example of LDM Masking Profiles

LDM masking profiles include attributes with characteristics of certain data. Profiles are used to connect data characteristics and masking algorithms with tables and fields in a data store that are going to be masked.

LDM Masking Procedures

Masking procedures are the components of the LDM solution to drive the process for updating data stores with new values. Procedure components include reference data, masking keying process, and the actual masking algorithms.

Masking Reference Data

Reference data in the context of LDM refers to a pre-populated collection of random, non-sensitive information which is used to replace sensitive data in data stores that require anonymization. Libelle provides a default reference database which contains random names, addresses, and other data used in the masking process. For first and last names, the reference database provides geographical regions. LDM regions provide, for example, names that are common in such regions to recreate a resemblance of the original data while retaining full anonymity.

ID	Description
SL	Russia and Slavic Countries
DE	Germany and Austria
US	The United States of America
FR	France, Andorra, Monaco, Reunion
CN	Japan, China and Southeast-Asia
IN	India, Bangladesh and Sri Lanka
RL	Persian and Arabic-Speaking Countries
…	…

Table 4 – Example of Selected LDM Masking Regions

The LDM reference database is one source that can be utilized by the masking algorithms. Aside from extending the reference database, users may provide other sources, such as a previously masked data store, or a reference database that is created at runtime. LDM creates relationships to the replacement data during the prepare phase in a mapping process as outlined in the diagram below.

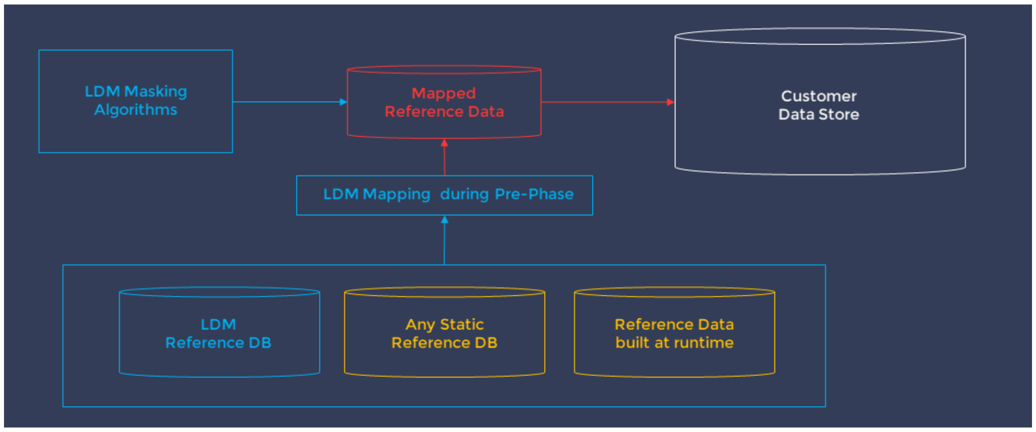


Figure 4 – Masking Reference Data Sources

Masking Keying Process

Masking keying process refers to the LDM procedures involved in creating a masking key that is used for each masking run and deriving a consistent, secure lookup table. The lookup table is built at runtime and used in a way that a unique masking key always masks the same input values to the same output values.

At the beginning of the masking process, the administrator provides a masking key, typically an alphanumeric passphrase. A proprietary SHA2-based algorithm derives a unique nummap.bin file which is never identical for two different keys. The nummap.bin file is essentially a lookup table where each unique input value provides one unique output value and the pairing process is bijective. Result is that each unique input value with the same masking key will always lead to the same output value, allowing consistent masking across the data store for secondary indices or other data that needs to be masked consistently.

The algorithm can be used for names as illustrated below. It can also be used for number values, in particular if certain integer values serve as secondary index references to another field in the same data store, or if multiple data stores contain the same data. The result is completely randomized data that is still fully consistent with the data store and across other connected data stores.

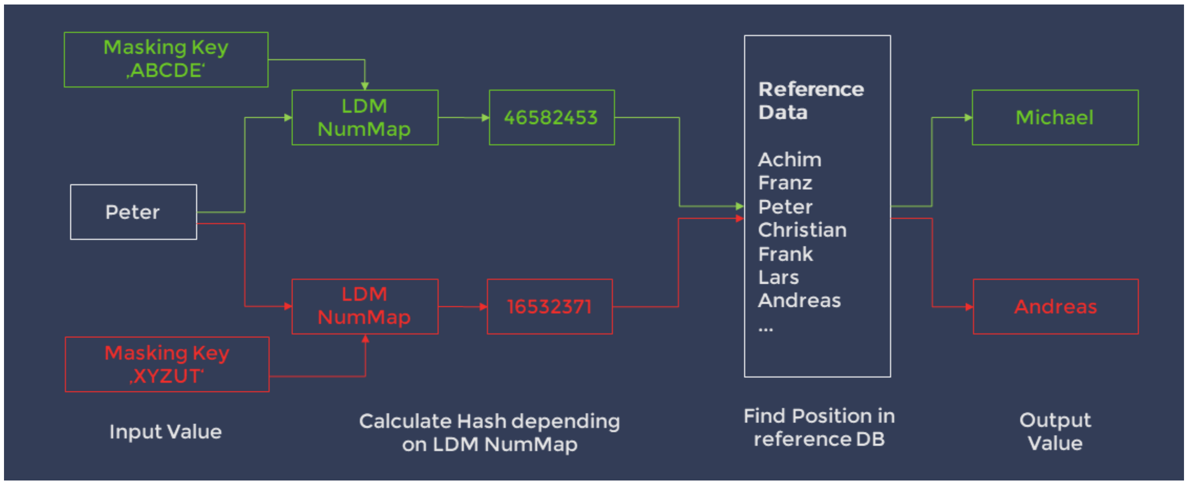


Figure 5 – Consistent Masking via Masking Keys

LDM masking keys allow the same input value to mask reliably to the same output value. With that, Libelle produces consistent masking results within and across data stores.

Standard or Custom Algorithms

LDM comes with numerous standard masking algorithms, and users can adjust existing algorithms, or develop their own. Generally, all LDM default algorithms ignore input values defined as ‘White Space’, ‘Empty String’, and ‘Nil’. Masking algorithms are extended on an ongoing basis and there are 40+

algorithms available. A variety of options, such as conditional masking or masking groups, accommodate specific requirements for post-masking data consistency.

There are 40+ default masking algorithms available, with various options for additional parametrizations such as setting ranges for output values, conditional masking, or groups that are masked together (e.g. a masked country code and country) to ensure consistency.

Generic Number Masking

The following are examples of number masking algorithms available from the LDM repository. There are additional algorithms available.

LDM Algorithm	Description	Example	
		Input	Output
aNumber	Create numbers using the defined masking key to ensure bijectivity (each input value has exactly one output value). With that it can be used to anonymize values that must be unique. Non-digit characters will be ignored.	1234 S-L-1234 A1B2C34 0012 BBBB	0356 S-L-0356 A0B3C56 8510 BBBB
aNumberRange	Same as aNumber, but only works if complete value contains digits. Will retain leading 0.	0001234 01234 000000	0001745 01745 000000
aNumberZero	Anonymize value only if at least one of digit values is not equal to zero. Non-digit values will be never changed.	123	348
		1.0.2-A	7.1.0-A
		0	0
		0.00	0.00
		4.000	8.102
aSerial	Anonymize - ignore leading zeros and anonymize like aNumber the rest of value.	00123	00301
		0012S3	0030S1
		1234	0356
		0	0
…	…	…	..

Table 5 – Number Masking Algorithm

Generic Alphanumeric Masking Algorithms

The following are example of alphanumeric algorithms available from the LDM repository.

LDM Algorithm	Description	Example

		Input	Output
aAlphabetic	Anonymize only Latin characters in ASCII between. Numbers will be ignored.	S-L-1234	M-P-1234
		ABB	JVV
		1234	1234
aAlphanumeric	Combines aNumber and aAlphabetic, so that both numbers and characters are masked.	S-L-1234	M-P-0356
		Alp12	Khj79
		1234	0356
aAlphanumeric_UTF8	Based starting Unicode value values are masked with same Unicode Area. Digits will be anonymized like aNumber.	广文字第 03086	巾女八元 32124
		073 Ἀ Δ'ΘΕΣΣΑΛΟΝΙ	004 ὢA'ΑνάάΘΰΖζη
		装文字第 081928	弓女八元 255451
		عبد الملك	غيؤ بؤه

Table 6 – Alphanumeric Masking Algorithm

Masking Algorithms for Names

The LDM first names algorithm takes gender into consideration and maintains a given gender. Name masking algorithms are surjective by default (output values may repeat with the same input values). Both first name and last name take the Libelle reference database into consideration where names are masked based on a configured region. Names can be masked separately by first name and last name, or input can be fields with full names following the same rules and provide exactly one first name and one last name from the reference database, even if there are additional words in the input name. LDM calculates digit sums from the input name and looks up an output name. In the second example below, we intentionally provided a surjective example where a last name may repeat. This is different from number masking which is designed for bijective masking.

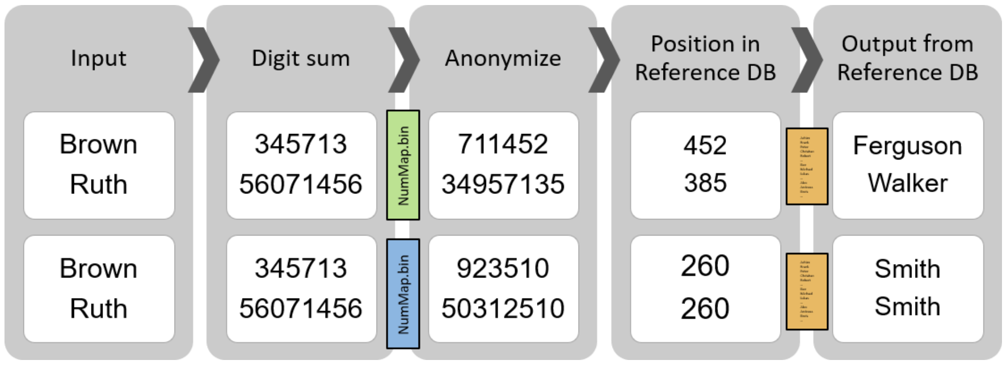
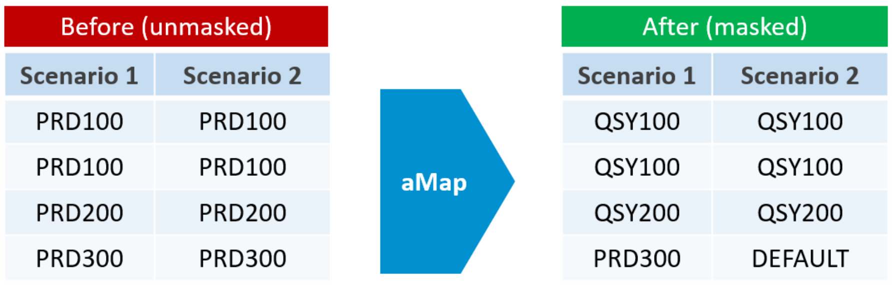


Figure 6 – Example for Masking Names with LDM

Masking via Mapping Input Values with Defined Output Values

The LDM aMap algorithm replaces pre-defined input values in a data store with pre-defined output values. As such, it is technically not an anonymization algorithm but is often used in the context of data masking to adjust data structures post-masking. The new values come as single or multiple columns from a configured LDM reference file. If input values do not exist, they can be ignored (Scenario 1), or replaced by a default value (Scenario 2).



Before (unmasked)	
Scenario 1	Scenario 2
PRD100	PRD100
PRD100	PRD100
PRD200	PRD200
PRD300	PRD300

After (masked)	
Scenario 1	Scenario 2
QSY100	QSY100
QSY100	QSY100
QSY200	QSY200
PRD300	DEFAULT

Figure 7 – aMap Algorithm

Masking Groups of Fields Across Multiple Columns

The aReference algorithm allows users to anonymize multiple fields in a data store which logically belong together with new values from LDM reference data. The aAddress-masking algorithm is a specific implementation of this algorithm where anonymized postal codes match an anonymized city, and that city matches a state, etc. aReference is a generic implementation of the same logic where a reference file can contain any logically connected fields that span across multiple columns.

Conditional Data Masking

LDM provides a variety of options for conditional data masking. A basic masking configuration may include all fields from a specific database column identified by column name. By default, all fields will be masked with a specific algorithm. Conditional masking comes into play if each field needs to meet certain criteria to be masked. Criteria for conditions can be a simple “if clause” contained within the algorithm or a lookup query to other areas of a data store. One such example of conditional masking implementation is a lookup query to ascertain if a Manufacturers’ Material Description Field is part of ITAR-restricted materials which is defined in another table. If the material is ITAR-restricted, it would be masked; otherwise it would not.

Conditional data masking matches user and data store specific requirements for looking up criteria that are statically given or determined at run-time.

Other Basic Masking Algorithms

In addition to specific examples outlined previously, additional masking algorithms are available. Each algorithm may have subsets and follows certain rules and patterns to ensure meaningful output data.

LDM Algorithm	Description
aEmail	Anonymizes email addresses. LDM reference database has generic corporate and public domains which are appended with a .anon postfix. First and last names, if given, are logically separated if possible and algorithms for first and last name are used to recreate the username part of the email.
aDate	Determines common date formats and anonymize, returning the same structure that was provided. Range for years can be provided so that output values stay within given ranges.
aDateFormatted	Similar to aDate, but with more options to define specific date formats, consideration for synchronized dates, and accommodates more stringent format requirements.
aUrl	Algorithm to anonymize URI's which are replaced with URI's from Libelle's reference database.
aConstant	Anonymize input values by replacing with a constant output value.
aEmpty	Anonymize input values by setting values to whitespace. If other characters are required instead, aConstant should be used.
aOrganization	Anonymizes organizations' names using the LDM reference database.
aCreditCard	Anonymize credit card numbers by creating random numbers and ensuring check digits and other validation elements are still valid post-masking.
aIban	Anonymize IBAN numbers. Only valid input values will be anonymized. LDM executes IBAN algorithms country-specific, so that the country-code stays unchanged, but the routing and bank account information are fully anonymized while maintaining a valid IBAN code.
aAddress	Anonymize addresses using data sets from the reference database. Data sets match selected fields such as country, postal code, and city name to generate a correct new address.

Table 7 – Other Basic Masking Algorithms

Basic masking algorithms cover the most common data masking requirements to anonymize fields such as names, addresses, dates, credit cards, bank information, etc.

Data Masking in Action

Masking Configurations

After installing LDM, users can then define one or multiple masking configurations. Each configuration has a dedicated data store, defined masking fields, and defined masking algorithms. All configuration data is stored and managed on a single LDM master server as outlined in diagram below.

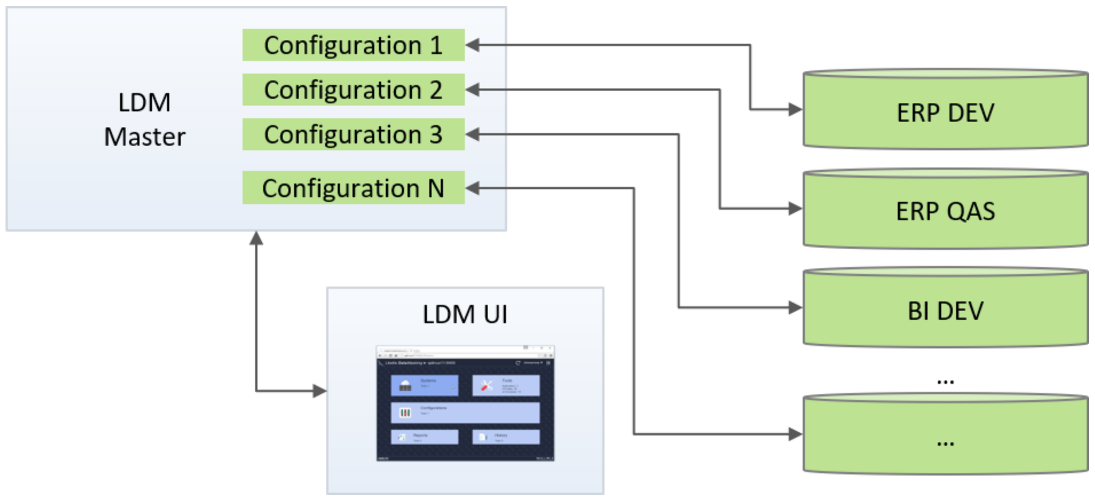


Figure 8 – Configuration Management

Besides pre-configured masking profiles and masking procedures, Libelle also provides pre-configured templates for certain applications, such as human resource data in SAP® ERP environments.

Masking Execution

During development of LDM, Libelle spent a significant portion of its R&D resources into optimizing and accelerating the masking execution. Data masking can be an expensive process from a compute and storage perspective. Updating hundreds of millions or sometimes billions of rows is not unusual in masking requirements.

In addition to heavy use of database indexes, LDM has sophisticated workflows for sending bulk-updates directly to databases using native database connectors or standard SQL. The masking process was designed to fully utilize scale-up concepts so that update jobs can spread across multiple CPUs. Even fairly small environments with only a handful of CPUs can easily reach 100k+ updates per second.

LDM masks directly on a database level, running updates in parallel, utilizing indexes, etc.
This allows to anonymize millions or billions of rows of data in a reasonable execution period.

Summary of Key Features

Below are some of the key features and functionalities of Libelle **Data***Masking*:

- LDM replaces original data in non-production systems with random data; sensitive data is fully protected as it is completely removed.
- Data remains fully usable with all characteristics such as real-appearing names, retained character encodings, retained patterns such as account numbers, etc.
- LDM provides pre-configured masking profiles such as name fields, fields for account numbers, country-specific addresses and locations, etc.
- LDM provides pre-configured masking algorithms such as replacing names with Libelle names generated from a reference database, shifting date of birth within limited ranges, or credit cards with valid check digits.
- For SAP applications, LDM utilizes the SAP Data Dictionary to support users in identifying sensitive fields.
- Upon masking execution, user provides a unique masking key (i.e. passphrase). Masking keys ensure consistent application of masking algorithms, i.e. the same input with same masking key always produces the same output. This provides referential integrity of the data model without the risk associated with reversibility.

Key Benefits

- **One-Stop Solution –** Enterprise-wide data masking solution
- **Performance –** High-performing and scalable algorithm reduces time to mask large data sets
- **Robust –** Well-designed software architecture provide enterprise-level stability
- **Modern –** HTML based UI for masking users
- **Secure –** Role-based LDAP authorization and authentication
- **Dedicated –** Dedicated solution to data obfuscation
- **Consistent –** Masking key allows consistent masking within and across data stores
- **Simple to Implement** –Pre-configured profiles, algorithms, and application templates

Libelle provides one of the most mature and complete data masking solutions on the enterprise market.

Disclaimer

Rev 1.2 / Feb 2020