DELPHIX

Delphix Masking User Guide

January 2019

Delphix Masking Engine API Cookbook
You can find the most up-to-date technical documentation at:
docs.delphix.com The Delphix Web site also provides the latest product updates.
If you have comments about this documentation, submit your feedback to:
infodev@delphix.com

Delphix Corp.
1400 Seaport Blvd, Suite 200
Redwood City, CA 94063

Introduction to Delphix Masking

Challenge

With data breach incidents regularly making the news and increasing pressure from regulatory bodies and consumers alike, organizations must protect sensitive data across the enterprise. Contending with insider and outsider threats while staying compliant with mandates such as HIPAA, PCI, and GDPR is no easy task—especially as teams simultaneously try to make their organizations more agile.

To tackle the problem of protecting sensitive information, companies are increasingly scrutinizing the tools they've deployed. Instead of reactive perimeter defenses, security minded organizations must focus on proactively protecting the interior of their systems: their data. Moreover, while mainstay approaches such as encryption may be effective for securing data-in-motion or data resident in hard drives, they are ill-suited for protecting non-production environments for development, testing, and reporting.

Solution

The masking capability of the Delphix Dynamic Data Platform represents an automated approach to protecting non-production environments, replacing confidential information such as social security numbers, patient records, and credit card information with fictitious, yet realistic data.

Unlike encryption measures that can be bypassed through schemes to obtain user credentials, masking irreversibly protects data in downstream environments. Consistent masking of data while maintaining referential integrity across heterogeneous data sources enables Delphix masking to provide superior coverage compared to other solutions—all without the need for programming expertise. Moreover, the Delphix Dynamic Data Platform seamlessly integrates masking with data delivery capabilities, ensuring the security of sensitive data before it is made available for development and testing, or sent to an offsite data center or the public cloud.

Delphix Masking is a multi-user, browser-based web application that provides complete, secure, and scalable software for your sensitive data discovery, masking and tokenization needs, while meeting enterprise-class infrastructure requirements. The Delphix Dynamic Data Platform has several key characteristics to enable your organization to successfully protect sensitive data across the enterprise:

- End-to-End Masking — The Delphix platform automatically detects confidential information, irreversibly masks data values, then generates reports and email notifications to confirm that all sensitive data has been masked.

- Realistic Data — Data masked with the Delphix platform is production-like in quality. Masked application data in non-production environments remains fully functional and realistic, enabling the development of higher-quality code.

- Masking Integrated with Virtualization — Most masking solutions fail due to the need for repeated, lengthy batch jobs for extracting and masking data and lack delivery capabilities for downstream environments. The Delphix Dynamic Data Platform seamlessly integrates data masking with data virtualization, allowing teams to quickly deliver masked, virtual data copies on premises or into private, public, and hybrid cloud environments.

- Referential Integrity — Delphix masks consistently across heterogeneous data sources. To do so, metadata and data is scanned to identify and preserve the primary/foreign key relationships between elements so that data is masked the same way across different tables and databases.

- Algorithms/Frameworks — Seven algorithm frameworks allow users to create and configure algorithms to match specific security policies. Over twenty five out-of-the-box, preconfigured algorithms help businesses mask everything from names and addresses to credit card numbers and text fields. Moreover, the Delphix platform includes prepackaged profiling sets for healthcare and financial information, as well as the ability to perform tokenization: a process that can be used to obfuscate data sent for processing, then reversed when the processed data set is returned.

- Ease of Use — With a single solution, Delphix customers can mask data across a variety of platforms. Moreover, businesses are not

required to program their own masking algorithms or rely on extensive administrator involvement. Our web-based UI enables masking with a few mouse clicks and little training.

- Automated discovery of sensitive data — The Delphix Profiler automatically identifies sensitive data across databases and files, the time-consuming work associated with a data masking project is reduced significantly.

High Level Platform Architecture

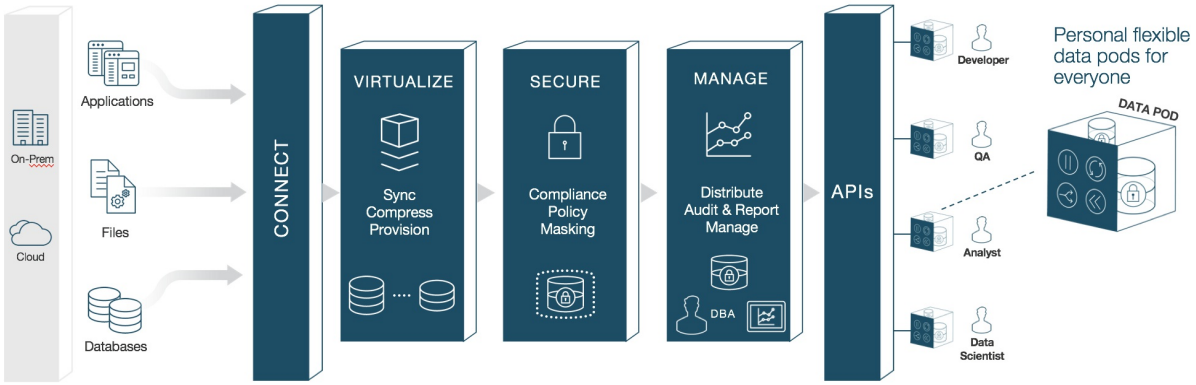
The Delphix Dynamic Data Platform is made up of 4 main services each of which play a very important part in delivering fresh secure data to anybody that needs it. These include:

- Virtualize — Delphix compresses the data that it gathers, often to one-third or more of the original size. From that compressed data footprint, Delphix virtualizes the data and allows operators to create lightweight, virtual data copies. Virtual copies are fully readable/writable and independent. They can be spun up or torn down in just minutes. And they take up a fraction of the storage space of physical copies -- 10 virtual copies can fit into the space of one physical copy.

- Identify and Secure — The Delphix platform continuously protects sensitive information with integrated data masking. Masking secures confidential data -- names, email addresses, patient records, SSNs -- by replacing sensitive values with fictitious, yet realistic equivalents. Delphix automatically identifies sensitive values then applies custom or predefined masking algorithms. By seamlessly integrating data masking and provisioning into a single platform, Delphix ensures that secure data delivery is effortless and repeatable.

- Manage — Data operators can now quickly provision secure data copies -- in minutes -- to users in their target environments. The Delphix platform serves as a single point of control to manage those copies. Data operators maintain full control and visibility into downstream environments. They can easily audit, monitor, and report against access and usage.

- Self Service — Provides developers, testers, analysts, data scientists, or other users with controls to manipulate data at-will. Users can refresh data to reflect the latest state of production, rewind environments to a prior point in time, bookmark data copies for later use, branch data copies to work across multiple releases, or easily share data with other users.



How Delphix Identifies Sensitive Data

Our platform helps you quickly identify your organization's sensitive data. This sensitive data identification is done using two different methods, column level profiling and data level profiling.

Column Level Profiling

Column level profiling uses REGEX expressions to scan the column names (metadata) of the selected data sources. There are several dozen pre-configured profile expressions (like the one below) designed to identify common sensitive data types (SSN, Name, Addresses, etc). You also have the ability to write/import your own profile expressions.

First Name Expression	<([A-Z][A-Z0-9]*)\b[^>]*>(.*?)</\1>

Data Level Profiling

Data level profiling also uses REGEX expressions, but to scan the actual data instead of the metadata. Similar to column level profiling, there are sefveral dozen pre-configured expressions (like the one below) and you can write/import your own.

Social Security Number Expression	<([A-Z][A-Z0-9]*)\b[^>]*>(.*?)</\1>

For both column and data level profiling, when data is identified as sensitive, Delphix recommends/assigns particular algorithms to be used when securing the data. The platform comes with several dozen pre-configured algorithms which are recommended when the profiler finds certain sensitive data.

How Delphix Secures Your Sensitive Data

Delphix strives to make available multiple methods for securing your data, depending on your needs. The two secure methods Delphix currently supports are masking (anonymization) and tokenization (pseudonymization).

Masking

Data masking secures your data by replacing values with realistic yet fictitious data. Seven out-of-the-box algorithm frameworks help businesses mask everything from names and social security numbers to images and text fields. Algorithms can also be configured or customized to match specific security policies.

Before Masking	After Masking
Elon Musk	Jeff Bezos

Tokenization

Tokenization uses reversible algorithms so that the data can be returned to its original state. Tokenization is a form of encryption where the actual data – such as names and addresses – are converted into tokens that have similar properties to the original data (text, length, etc.) but no longer convey any meaning.

Before Tokenizing	After Tokenizing
226-74-3756	256-37-7426

What's New for Masking

Synchronizing Masking Jobs & Universal Settings Across Engines

In 5.2 we introduced the ability to synchronize Masking Algorithms between engines to ensure consistent masking, regardless of the engine executing the masking. In 5.3 are expanding the list of syncable objects to include:

- Masking Jobs
- Connectors
- Rulesets
- Domains
- File Formats

The sync of objects is possible through improvements to several sync API endpoints, including:

- GET /syncable-objects[?object_type=]
- POST /export
- POST /export-async
- POST /import
- POST/import-async

This expansion of syncable objects ensures that users can sync their Masking Jobs and all the objects necessary for that masking job to execute successfully - regardless of the masking engine it lives on, allowing for easier scaling of Delphix Masking across the enterprise. Please see Managing Multiple Masking Engines for more details.

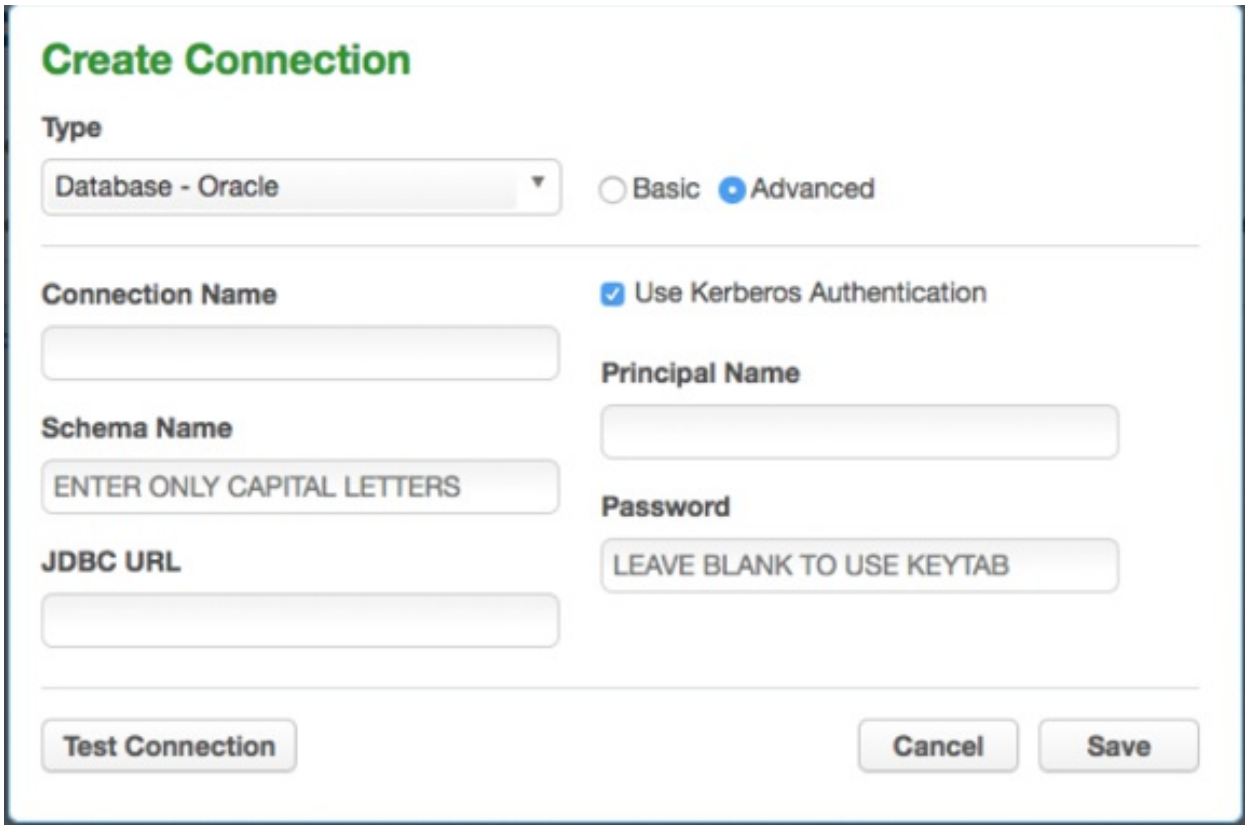
Support for Kerberized Connections

In 5.2.4 we added support for Kerberos for our Oracle Masking Connector. In 5.3 we have expanded the list of connectors that support Kerberos to:

- SQL Server
- Sybase

To enable Kerberized connectors your engine must be configured properly and you must configure your masking Connectors for Kerberos. Kerberos can be enabled by going to the

Advanced mode on Oracle, SQL Server and Sybase. Please see Managing Connectors for more details.



New API Endpoints

In 5.2 we released an all-new set of API endpoints allowing for the automation of many masking workflows. In 5.3 we have expanded this list of API endpoints around Algorithms, Users, Roles, File Upload, System Information, Login, Rulesets, and Connector. Below are the net new API endpoints:

Group	Endpoints	Description
Algorithms	POST /algorithms	Create algorithm
	DELETE /algorithms/{algorithmName}	Delete algorithm by name
	GET /algorithms/{algorithmName}	Get algorithm by name
	PUT /algorithms/{algorithmName}	Update algorithm by name

	PUT /algorithms/{algorithmName}/randomize-key	Randomize key by name
Users	GET /users	Get all users
	POST /users	Create user
	DELETE /users/{userId}	Delete user by ID
	GET /users/{userId}	Get user by ID
	PUT /users/{userId}	Update user by ID
Roles	GET /roles	Get all roles
	POST /roles	Create role
	DELETE /roles/{roleId}	Delete role by ID
	GET /roles/{roleId}	Get role by ID
	PUT /roles/{roleId}	Update role by ID
Rulesets	PUT /database-rulesets/{databaseRulesetId}/bulk-table-update	Update the rule set’s tables
	PUT /database-rulesets/{databaseRulesetId}/refresh	Refresh the rule set
Connectors	POST /database-connectors/{databaseConnectorId}/test	Test a database connector
	POST /database-connectors/test	Test an unsaved database connector
	POST /file-connectors/{fileConnectorId}/test	Test a file connector
	POST /file-connectors/test	Test an unsaved file connector
Async Tasks	GET /async-tasks	Get all asyncTasks
		Get asyncTask by

	GET /async-tasks/{asyncTaskId}	ID
	PUT /async-tasks/{asyncTaskId}/cancel	Cancel asyncTask by ID
File Upload/Download	DELETE /file-uploads	Delete all file uploads
	POST /file-uploads	Upload file
	GET /file-downloads/{fileDownloadId}	Download file
System Information	GET /system-information	Get version, etc.
Login/Logout	PUT /logout	User logout
Executions	GET /execution-components	Status for a table, file, or Mainframe file

In addition to the net new API endpoints, we have improved pre-existing API endpoints. Some of the improvements include:

- Addition of DB2 iSeries & Mainframe to connector endpoints.
- Addition of Kerberos configuration on Oracle, SQL Server & Sybase connectors
- Ability to have ruleset refresh drop tables
- Support for XML file types
- Addition of dataType to column metadata

For more information please on Delphix Masking APIs please see API documentation. Please note that the previous generation of Masking APIs (commonly referred to as V4) is EOL and no longer supported in this release. All users are encouraged to migrate to the V5 APIs.

Prerequisites

This section will detail the hardware/software requirements needed to deploy the Delphix Engine with the Masking service. The Delphix Engine is a self-contained operating environment and application that is provided as a Virtual Appliance. Our Virtual Appliance is certified to run on a variety of platforms including VMware, AWS, and Azure.

The Delphix Engine should be placed on a server where it will not contend with other VMs for network, storage or other compute resources. The Delphix Engine is a CPU and I/O intensive application, and deploying it in an environment where it must share resources with other virtual machines, can significantly reduce performance.

Delphix Masking and Delphix Virtualization should never be run inside the same virtual machine. Always use seperate, dedicated Delphix Engines for Masking and Virtualization.

Client Web Browser

The Delphix Engine's graphical interface can be accessed from a variety of different web browsers. The Delphix Engine currently supports the following web browsers:

- Microsoft Internet Explorer 10.0 or higher
- Mozilla Firefox 35.0 or higher
- Chrome 40 or higher

!!! tip "TIP - Microsoft Internet Explorer"
Make sure that Internet Explorer is not configured for compatibility mode.

VMware Virtual Platform

This section covers the virtual machine requirements for installation of a dedicated Delphix Masking Engine on the VMware Virtual platform.

VMWare Platform

The Delphix Engine can be run on several version of VMware ESX/ESXI (see support matrix).

Virtual CPUs

The minimum amount of virtual CPUs is 8v CPUs. CPU resource shortfalls can occur under high I/O throughput conditions. Also, CPU reservation is strongly recommended for the Delphix VM, so that Delphix is guaranteed the full complement of vCPUs even when resources are overcommitted.

Virtual Memory

The minimum amount of virtual memory is 16GB vRAM but we highly recommend 32GB or higher. The masking service on the Delphix Engine uses its memory to process database and file blocks. More memory can sometimes improve performance. Memory reservation is a requirement for the Delphix VM.
Overcommitting memory resources in the ESX server will significantly impact the performance of the Delphix Engine. Reservation ensures that the Delphix Engine will not stall while waiting for the ESX server to page in the engine's memory.

!!! tip "TIP - Do not allocate all memory to the Delphix Engine"

```
Never allocate all available physical memory to the Delphix VM. You must set aside memory f

For example, when running on an ESX Host with 512GB of physical memory, allocate no more th
```

Delphix Engine System Disk Storage

The minimum recommended storage on the Delphix Engine System Disk is 300GB. The System disk may need to be substantially larger if bulk logging will be enabled. The actual size will depend on the data being masked. The VMDK for the Delphix Engine system disk storage is often created in the same VMFS volume as the Delphix VM definition. In that case, the datastore must have sufficient space to hold the Delphix VM configuration, the VMDK for the system disk, and a paging area if a memory reservation was not enabled for the Delphix Engine.

AWS EC2 Platform

This section covers the virtual machine requirements for installation of a dedicated Delphix Masking Engine on Amazon's Elastic Cloud Compute

(EC2) platform.

For best performance, the Delphix Masking Engine and all database servers should be in the same AWS region.

Instance Types

The Delphix Engine can run on a variety of different instances, including large memory instances (preferred) and high I/O instances. The Delphix Engine most closely resembles a storage appliance and performs best when provisioned using a storage optimized instance type. We recommend the following large memory and high I/O instances:

Large Memory Instances (preferred)	High I/O Instances (supported)
• r4.2xlarge • r4.4xlarge • r4.8xlarge • r3.2xlarge • r3.4xlarge • r3.8xlarge	• i3.2xlarge • i3.4xlarge • i3.8xlarge • i2.2xlarge • i2.4xlarge • i2.8xlarge

Larger instance types provide more CPU, which can prevent resource shortfalls under high I/O throughput conditions. Larger instances also provide more memory, which the Delphix Engine uses to cache database blocks. More memory will provide better read performance. For more information please refer to, Virtual Machine Requirements for AWS EC2 Platform.

Network Configurations

You must deploy the Delphix Engine and all database or file hosts in a VPC network to ensure that private IP addresses are static and do not change when you restart instances. When adding environments to the Delphix Engine, you must use the host's VPC (static private) IP addresses.

The EC2 Delphix instance must be launched with a static IP address; however, the default behavior for VPC instances is to launch with a dynamic public IP address – which can change whenever you restart the instance. If you're using a public IP address for your Delphix Engine, static IP addresses can only be achieved by using assigned AWS Elastic IP
Addresses.

!!! tip "TIP - Port Configuration"
The default security group will only open port 22 for secure shell (SSH) access. You must modify the security group to allow access to all of the networking ports used by the Delphix Engine and the various source and target engines. See General Network and Connectivity Requirements for information about specific port configurations.

EBS Configurations

All attached storage devices must be EBS volumes. Delphix does not support the use of instance store volumes. Because EBS volumes are connected to EC2 instances via the network, other network activity on the instance can affect throughput to EBS volumes. EBS optimized instances provide guaranteed throughput to EBS volumes and are required (for instance types that support it) in order to provide consistent and predictable storage performance.

Use EBS volumes with provisioned IOPs in order to provide consistent and predictable performance. The number of provisioned IOPs depends on the estimated IO workload on the Delphix Engine. Provisioned IOPs volumes must be configured with a volume size at least 30 GiB times the number of provisioned IOPs. For example, a volume with 3,000 IOPS must be configured with at least 100 GiB.

I/O requests of up to 256 kilobytes (KB) are counted as a single I/O operation (IOP) for provisioned IOPs volumes. Each volume can be configured for up to 4,000 IOPs.

General Storage Configurations

The minimum recommended storage on the Delphix Engine System Disk is 300GB. The System disk may need to be substantially larger if bulk logging will be enabled. The actual size will depend on the data being masked.

Add storage when storage capacity approaches 30% free. Keep all EBS volumes the same size. Add new storage by provisioning new volumes of the same size.

Use at least 3 EBS volumes to maximize performance. This enables the Delphix File System (DxFS) to make sure that its file systems are always consistent on disk without additional serialization. This also enables the Delphix Engine to achieve higher I/O rates by queueing more I/O operations to its storage.

Azure Platform

This section covers the virtual machine requirements for installation of a dedicated Delphix Masking Engine on Microsoft's Azure cloud platform.

For best performance, the Delphix Masking Engine and all database servers should be in the same Azure Region.

Instance Types

The Delphix Engine can run on a variety of different Azure instances. The Delphix Engine most closely resembles a storage appliance and performs best when provisioned using a storage optimized instance type. We recommend the following memory and storage optimized instances:

- GS3 - 8 CPUs, 112GB, 16 TB (20,000 IOPS)
- GS4 - 16 CPUs, 244GB, 32 TB (40,000 IOPS)
- GS5 - 32 CPUs, 448GB, 64 TB (80,000 IOPS)

Network Configurations

Network Configuration for Delphix Engines running on the Azure Platform can be configured on the Azure Virtual Network (VNet). Delphix Engine and all the source and target environments must be accessible within the same virtual
network.

!!! tip "TIP - Port Configuration"
You must modify the security group to allow access to all of the networking ports used by the Delphix Engine and the various source and target engines. See General Network and Connectivity Requirements for information about specific port configurations.

Storage Configurations

When configuring storage for the Delphix Engine we recommend Azure

Premium Storage which uses solid-state drives (SSDs). Devices up to 1024 are supported with a total maximum of 64tb.

I/O requests of up to 256 kilobytes (KB) are counted as a single I/O operation (IOP) for provisioned IOPs volumes. IOPS vary based on storage size with a maximum of 5,000 IOPS.

Data Source Support

The Delphix Masking service supports profiling, masking, and tokenizing a variety of different data sources including distributed databases, mainframe, PaaS databases, and files. At a high level, Delphix Masking breaks up support for data sources into two categories:

- Dedicated Delphix Connectors: These are data sources that the Delphix Engine can connect to directly using built-in connectors that have been optimized to perform masking, profiling and tokenization.

- FEML Sources: FEML (File Extract Mask and Load) is a method used to mask and tokenize data sources that do not have dedicated Delphix Connectors. FEML uses existing APIs from data sources to extract the data to a file, masks the file, and then uses APIs to load the masked file back into the database.

Dedicated Delphix Connectors

The Delphix Engine has dedicated masking connectors for the following data sources:

- Distributed Database: DB2 LUW, Oracle, MS SQL, MySQL, SAP ASE (Sybase), PostgreSQL, MariaDB

- Mainframe/Midrange: DB2 Z/OS, DB2 iSeries, VSAM

- PaaS Database: AWS RDS Oracle, RDS PostgreSQL, RDS MYSQL, RDS MARIA DB, RDS MS SQL Server

- Files: Excel, Fixed Width, Delimited, XML

For a detailed view of all the versions, features, etc Delphix supports on each data source - see the sections below.

DB2 LUW Connector

Introduction

DB2 for Linux, UNIX and Windows is a database server product developed by IBM. Sometimes
called DB2 LUW for brevity, it is part of the DB2 family of database products. DB2 LUW is the
"Common Server" product member of the DB2 family, designed to run on the most popular
operating systems. By contrast, all other DB2 products are specific to a single platform.

Support Matrix

The Delphix DB2 LUW connector supports the following versions of DB2 LUW:

Version	Linux	Unix	Windows
9.1	Supported	Supported	Supported
9.5	Supported	Supported	Supported
9.7	Supported	Supported	Supported
9.8	Supported	Supported	Supported
10.1	Supported	Supported	Supported
10.5	Supported	Supported	Supported
11.1	Supported	Supported	Supported

Available Features

The DB2 LUW connector supports profiling and masking/tokenization features. Below is a list of which options are & are not available for jobs using the DB2 LUW connector:

	Feature	Availability
In-Place Masking Mode	Multi-Tenant	Available
	Streams / Threads	Available
	Bulk Update	Available
	Batch Update	Available

	Drop Indexes	Available
	Disable Trigger	Unavailable
	Disable Constraint	Unavailable
	Identity Column Support	Unavailable
On-The-Fly Masking Mode	Restart Ability	Available
	Truncate	Available
	Disable Trigger	Unavailable
	Disable Constraint	Unavailable
	Create Target	Available
Profiling	Multi-Tenant	Available
	Streams	Available

Oracle Connector

Introduction

Oracle Database (commonly referred to as Oracle RDBMS or simply as Oracle) is a multi-model database management system produced and marketed by Oracle Corporation.

Support Matrix

Version	Linux	Unix	Windows	AWS RDS
10g	Supported	Supported	Supported	Supported
11gR1	Supported	Supported	Supported	Supported
11gR2	Supported	Supported	Supported	Supported
12c	Supported	Supported	Supported	Supported
12cR2	Supported	Supported	Supported	Supported

18c	Supported	Supported	Supported	Supported

Available Features

The Oracle connector supports profiling and masking/tokenization features. Below is a list of which options are and are not available for jobs using the Oracle connector:

	Feature	Availability
In-Place Masking Mode	Multi-Tenant	Available
	Streams / Threads	Available
	Bulk Update	Available
	Batch Update	Available
	Drop Indexes	Available
	Disable Trigger	Available
	Disable Constraint	Available
	Identity Column Support	Available
On-The-Fly Masking Mode	Restart Ability	Available
	Truncate	Available
	Disable Trigger	Available
	Disable Constraint	Available
	Create Target	Available
Profiling	Multi-Tenant	Available
	Streams	Available

MS SQL Connector

Introduction

Microsoft SQL Server is a relational database management system

developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications—which may run either on the same computer or on another computer across a network (including the Internet).

Support Matrix

Version	Linux	Unix	Windows	AWS RDS
2005	N/A	N/A	Supported	Supported
2008	N/A	N/A	Supported	Supported
2008 R2	N/A	N/A	Supported	Supported
2012	N/A	N/A	Supported	Supported
2014	N/A	N/A	Supported	Supported
2016	Supported	N/A	Supported	Supported

Available Features

The MS SQL connector supports profiling and masking/tokenization features. Below is a list of which options are and are not available for jobs using the MS SQL connector.

	Feature	Availability
In-Place Masking Mode	Multi-Tenant	Available
	Streams / Threads	Available
	Bulk Update	Available
	Batch Update	Available
	Drop Indexes	Available
	Disable Trigger	Available
	Disable Constraint	Available
	Identity Column Support	Available
On-The-Fly Masking Mode	Restart Ability	Available

	Truncate	Available
	Disable Trigger	Available
	Disable Constraint	Available
	Create Target	Available
Profiling	Multi-Tenant	Available
	Streams	Available

PostgreSQL Connector

Introduction

PostgreSQL, often simply Postgres, is an object-relational database management system (ORDBMS) with an emphasis on extensibility and standards compliance. PostgreSQL is developed by the PostgreSQL Global Development Group, a diverse group of many companies and individual contributors. It is free and open-source, released under the terms of the PostgreSQL License, a permissive software license.

Support Matrix

Version	Linux	Unix	Windows	AWS RDS
9.2	Supported	Supported	Supported	Supported
9.3	Supported	Supported	Supported	Supported
9.4	Supported	Supported	Supported	Supported
9.5	Supported	Supported	Supported	Supported
9.6	Supported	Supported	Supported	Supported
10	Supported	Supported	Supported	Supported

Available Features

The PostgreSQL connector supports profiling and masking/tokenization features. Below is a list of which options are & are not available for jobs using the PostgreSQL connector:

	Feature	Availability
In-Place Masking Mode	Multi-Tenant	Available
	Streams / Threads	Available
	Bulk Update	Available
	Batch Update	Available
	Drop Indexes	Unavailable
	Disable Trigger	Unavailable
	Disable Constraint	Unavailable
	Identity Column Support	Available
On-The-Fly Masking Mode	Restart Ability	Unavailable
	Truncate	Available
	Disable Trigger	Available
	Disable Constraint	Available
	Create Target	Available
Profiling	Multi-Tenant	Available
	Streams	Unavailable

MySQL / MariaDB Connector

Introduction

MySQL is an open-source relational database management system (RDBMS). MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB. MySQL is now owned by Oracle Corporation.

MariaDB is a community-developed fork of the MySQL relational database management system intended to remain free under the GNU GPL. Development is led by some of the original developers of MySQL, who forked it due to concerns over its acquisition by Oracle Corporation.

A MySQL Connector may be used to connect to either a MySQL or MariaDB database instanace.

MySQL Support Matrix

Version	Linux	Unix	Windows	*AWS RDS**
5.5	Supported	Supported	Supported	Supported
5.6	Supported	Supported	Supported	Supported
5.7	Supported	Supported	Supported	Supported

MariaDB Support Matrix

Version	Linux	Unix	Windows	AWS RDS
10	Supported	Supported	Supported	Supported

Available Features

The MySQL connector supports profiling and masking/tokenization features. Below is a list of which options are and are not available for jobs using the MySQL connector:

	Feature	Availability
In-Place Masking Mode	Multi-Tenant	Available
	Streams / Threads	Available
	Bulk Update	Available
	Batch Update	Available
	Drop Indexes	Available
	Disable Trigger	Unavailable
	Disable Constraint	Unavailable
	Identity Column Support	Available
On-The-Fly Masking Mode	Restart Ability	Unavailable

	Truncate	Available
	Disable Trigger	Unavailable
	Disable Constraint	Unavailable
	Create Target	Available
Profiling	Multi-Tenant	Available
	Streams	Available

SAP ASE (Sybase) Connector

Introduction

SAP ASE (Adaptive Server Enterprise), originally known as Sybase SQL Server, and also commonly known as Sybase DB or Sybase ASE, is a relational model database server product for businesses developed by Sybase Corporation which became part of SAP AG.

Support Matrix

Version	Linux	Unix	Windows
15.03	Supported	Supported	Supported
15.5	Supported	Supported	Supported
15.7	Supported	Supported	Supported
16	Supported	Supported	Supported

Available Features

The SAP ASE (Sybase) connector supports profiling and masking/tokeniztion features. Below is a list of which options are and are not available for jobs using the SAP ASE connector:

	Feature	Availability
In-Place Masking Mode	Multi-Tenant	Available

	Streams / Threads	Available
	Bulk Update	Available
	Batch Update	Available
	Drop Indexes	Available
	Disable Trigger	Available
	Disable Constraint	Available
	Identity Column Support	Available
On-The-Fly Masking Mode	Restart Ability	Available
	Truncate	Available
	Disable Trigger	Available
	Disable Constraint	Available
	Create Target	Available
Profiling	Multi-Tenant	Available
	Streams	Available

DB2 Z/OS and iSeries Connectors

Introduction

DB2 for z/OS and iSeries are relational database management systems that run on IBM Z(mainframe) and IBM Power Systems.

Support Matrix

Version	z/OS	i-Series
7.1	N/A	Supported
7.2	N/A	Supported
7.3	N/A	Supported

9	Supported	N/A
10	Supported	N/A
11	Supported	N/A
VSAM	Supported	N/A

Available Features

The DB2 for z/OS and iSeries connectors support profilinng and masking/tokenization features. Below is a list of which options are and are not available for jobs using the DB2 and z/OS and iSeries connectors:

	Feature	Availability
In-Place Masking Mode	Multi-Tenant	Available
	Streams / Threads	Available
	Bulk Update	Available
	Batch Update	Available
	Drop Indexes	Unavailable
	Disable Trigger	Unavailable
	Disable Constraint	Unavailable
	Identity Column Support	Unavailable
On-The-Fly Masking Mode	Restart Ability	Unavailable
	Truncate	Available
	Disable Trigger	Unavailable
	Disable Constraint	Unavailable
	Create Target	Unavailable
Profiling	Multi-Tenant	Available
	Streams	Available

Files Connector

Introduction

Much of the time data will live outside of databases. The data can be stored in a variety of different formats including Fixed Width, Delimited, etc.

Support Matrix

File Type/Format	Support Level
Excel (.xls & .xlsx)	Supported
Fixed Width	Supported
Delimited	Supported
XML	Supported
JSON	Not Supported

Installation/First Time Setup

This section walks you step by step on how to download and install the Delphix Engine software onto your infrastructure (VMware, AWS EC2, or Azure).

Installing OVA on VMware

For detailed recommendations on hardware prerequisites for VMware, please see Getting Started - Prerequisites. Here are the steps to getting your OVA installed:

1. Download the OVA file from Delphix's Download site.
   Note, you will need a support login from your sales team or welcome letter. Navigate to "Virtual Appliance" and download the appropriate OVA. If unsure, use the HWv8_Standard type.

2. Login using the vSphere client to the vSphere server (or vCenter Server) where you want to install the Delphix Engine.

3. In the vSphere Client, click File.

4. Select Deploy OVA Template and then browse to the OVA file. Click Next.

5. Select a hostname for the Delphix Engine. This hostname will be used in configuring the Delphix Engine network.

6. Select the data center where the Delphix Engine will be located.

7. Select the cluster and the ESX host.

8. Select one (1) data store for the Delphix OS. This datastore can be thin-provisioned and must have enough free space to accommodate the 300GB comprising the Delphix operating system.

9. The Delphix VM Configuration Storage requires a minimum of 10GB. The VMFS volume should have enough available space to hold all ESX configuration and log files associated with the Delphix Engine.

The Delphix Engine system disk should be stored in a VMDK system drive. The VMFS volume where the .ova is deployed should therefore have at least 300GB of free space prior to deploying the .ova. The VMFS volume must be located on shared storage in order to use vMotion and HA features.

10. Select the virtual network you want to use. If using multiple physical NICs for link aggregation, you must use vSphere NIC teaming. Do not add multiple virtual NICs to the Delphix Engine itself. The Delphix Engine should use a single virtual network. For more information, see Optimal Network Architecture for the Delphix Engine.

11. Click Finish. The installation will begin and the Delphix Engine will be created in the location you specified.

12. Jump to “Activating the Masking Service” section below to learn how to activate the masking service now that you have the software installed.

Installing AMI on AWS EC2

For detailed recommendations on hardware prerequisites for AWS EC2, please see Getting Started - Prerequisites. Here are the steps to getting your AMI installed:

1. On the Delphix download site, click the AMI you would like to share and accept the Delphix License agreement. Alternatively, follow a link given by your Delphix solutions architect.

2. On the Amazon Web Services Account Details form presented:

    - Enter your AWS Account Identifier, which can be found here: https://console.aws.amazon.com/billing/home?#/account. If you want to use the GovCloud AWS Region, be sure to enter the ID for the AWS Account which has GovCloud enabled.

    - Select which AWS Region you would like the AMI to be shared in. If you would like the AMI shared in a different region, contact your Delphix account representative to make the proper arrangements.

3. Click Share. The Delphix Engine will appear in your list of AMIs in AWS momentarily.

4. Reference the Installation and Configuration Requirements for AWS/EC2 when deploying the AMI.

5. Jump to “Activating the Masking Service” section below to learn how to activate the masking service now that you have the software installed.

Installing VHD on AZURE

For detailed recommendations on hardware prerequisites for Azure, please see Getting Started - Prerequisites. Here are the steps to getting your VHD installed:

1. On the Microsoft Azure Marketplace, search for Delphix. Click GET IT NOW.

2. Reference the Installation and Configuration Requirements for the Delphix Engine in Azure when deploying the VHD.

3. Jump to “Activating the Masking Service” section below to learn how to activate the masking service now that you have the software installed.

Activating the Masking Service

Once you have installed your Delphix Engine, you will need to activate the masking service through the CLI and then set up your first administrator user.

To activate the Masking service via the CLI, do the following:

1. Connect to the CLI via SSH as sysadmin or with other system administrator credentials.

2. Start the Delphix Masking Engine with: system ; startMasking ; commit ; exit

3. Access the UI by navigating to http://<Delphix Engine IP or DNS name>:8282/masking.

4. Login as user Admin and password Admin-12.

5. Change the Admin password to a unique value for your installation.

    a. To change the password, go to the Admin tab.

    b. Click Users.

    c. Edit the Admin user.

    d. Change the Admin user's password.

Congratulations! You are now ready to start using the masking service!

Configuring Virtualization Service for Masked Provisioning

Introduction

During the VDB provisioning process, the Virtualization Engine can optionally run a masking job from the Delphix Masking Engine on the VDB. By default, the Virtualization Engine attempts to obtain a list of masking jobs from a Masking Engine on its localhost. It's possible to split the Virtualization Engine and Masking Engine apart on separate hosts. If the Virtualization Engine and Masking Engine are on different hosts, use these instructions to customize the host address, port number, and/or login credentials that the Virtualization Engine will use to contact the Masking Engine.

!!! warning "Important Validation Notices"

```
When using seperate Virtualization and Masking engines, ensure that the versions are compat

Old versions of the serviceconfig or any information associated with them are not tracked.

Delphix does not validate network availability between the two engines or any other hosts t
```

Instructions

If the Virtualization Engine and Masking Engine are on different hosts, use these instructions to customize the host address, port number, and/or login credentials that the Virtualization Engine will use to contact the Masking Engine.

!!! note

```
This does not alter the Delphix Masking Engine UI port. It is specific to coordinating comm
```

To change the Virtualization Engine's connection details for its Masking Engine:

1. Using a shell, login to the CLI using delphix_admin.

2. At the CLI root prompt, type maskingjob.
3. At the maskingjob prompt, type serviceconfig.
4. To list service configurations, type ls.
5. At the serviceconfig, type select `MASKING_SERVICE_CONFIG-1.
6. To view the configurations, type ls.
7. With this service config selected, enter update.
8. In the update mode, use the set command to modify the configuration. For example, type set port=[YOUR DESIRED PORT NUMBER] to change the port number.
9. Commit the change by typing commit.
10. Type ls to confirm the configurations.
11. Type exit to exit the CLI.

Users and Roles

The Delphix Masking Service has a flexible and robust users and roles system that allows you to give users fine grain privileges over what environments they have access to and what tasks they can and can not perform.

What are Roles?

A defined role is what is used to give a certain user privileges over certain environments and tasks. Roles can be defined by selecting a subset of actions that can be taken on certain objects.

Actions

When defining a role, you can select one or more of the following actions for the role to be able to perform:

- View: Be able to view the object and important information about the object.
- Add: Be able to add an instance of an object.
- Update: Be able to update/edit an instance of an object.
- Delete: Be able to delete an instance of an object.
- Copy: Be able to create a copy of an object.
- Export: Be able to export an object from a Delphix Engine.
- Import: Be able to import an exported object into a Delphix Engine

Please note that not all of these actions are available for all objects in the masking service.

Objects

When defining a role, permission to perform the above actions can be defined on a per object basis. These objects include:

General	Jobs	Settings
Environment	Profile Job	Domains
Connection	Masking Job	Algorithms
Ruleset	Scheduler	Profiler
Inventory		Profile Set
		Mapping
		File Format
		Users

Please see Delphix Masking Terminology for definitions of these objects.

Adding A Role

To add a role follow these steps:

1. Login into the Masking Engine and select the Settings tab.

2. Click the Add Roles button.

3. Enter a Role Name. The far-left column lists the items for which you can set privileges.

4. Select the check boxes for the corresponding privileges that you want to apply. If there is no check box, that privilege is not available. For example, if you want this role to have View, Add, Update, and Run privileges for masking jobs, select the corresponding check boxes in the Masking Job row.

5. When you are finished assigning privileges for this Role, click Submit.

Recommended Roles

While every organization will differ in what users and roles they define, we have found that the following roles are common/popular:

- Analyst role — Can profile data and update inventories (but not create environments or connections)
- Developer role — Can create masking jobs and view reports
- Operator role — Can execute jobs (but cannot update inventories)
- Application owner role — Can define connections

!!! note "NOTE - One Role per User"
Please note that each defined user can only have one role assigned to them.

What are Users?

Once you have your roles defined, it is time to create users with those roles. We highly recommend creating independent users for each individual who will have access to the masking service.

Adding a User

To create a new user follow these steps:

1. Login into the Masking Engine and select the Admin tab.
2. Click Add User at the upper right of the Users screen.
3. You will be prompted for the following information:
    - First Name — The user's given name
    - Last Name — The user's surname
    - User Name — The login name for the user
    - Email — The user's e-mail address (mailable from the Delphix Masking Engine server for purposes of job completion e-mail messages)

- Password — The password that the Delphix Masking Engine uses to authenticate the user on the login page. The password must be at least six characters long, and contain a minimum of one uppercase character, one wild character (!@#$%^&*), and one number.

- Confirm Password — Confirm the password with double-entry to avoid data entry error.

- Administrator — (Optional) Select the Administrator check box if you want to give this user Administrator privileges. (Administrator privileges allow the user to perform all Delphix Masking Engine tasks, including creating and editing users in the Delphix Masking Engine.) If you select the Administrator check box, the Roles and Environments fields disappear because Administrator privileges include all roles and environments.

- Role — Select the role to grant to this user. The choices here depend on the custom roles that you have created. You can assign one role per user name.

- Environment — Enter as many environments as this user will be able to access. Granting a user access to a given environment does not give them unlimited access to that environment. The user's access is still limited to their assigned role.

4. When you are finished, click Save.

Delphix Masking Terminology

Before getting started with the Delphix Masking Engine, an overview of universal terms and concepts will build and unify how different masking components come together. The following provides a brief overview of the key concepts within the masking service.

High Level Concepts

These concepts are the high level concepts users run into.

Term	Definition
Application	An Application is a tag that is assigned to one or more environments. We recommend using an application name that is the same as the application associated with the environments.
Connector	Connectors are any set of data (database, file, etc) that have been connected to the Delphix Data Platform. These data sources can be physical or virtualized data sources.
Domain	A domain represents a correlation between various sensitive data categories (social security numbers) and the way it should be secured.
Environment	An environment is a construct that can be used to describe a collection of masking jobs associated with a group of data sources.
In-place	In-place masking is 1 of 2 procedures that can be used to apply masking algorithms to a data source. By choosing the In-place option, Delphix will read data from the data source, secure the data in the Engine and then update the data source with the secure data.
On-the-fly	On-the-fly masking is the second procedure that can be used to apply masking algorithms to a data source. By choosing the On-the-fly option, Delphix will read data from the data source, secure the data in the Engine and then place the secure data in a target source (different from the location of the original data source).
Inventory	An inventory describes all of the data present in a particular data source and defines the methods which will be used to secure it. Inventories typically include the table name, column name, the data

	classification, and the chosen algorithm.
Profile	Profiling uses a variety of different methods to classify data in a data source into different categories. These categories are known as domains. The profile process also assigns recommended algorithms for securing the data based on the the domain.
Ruleset	A rule set is group of tables or flat files within a particular data source that a user may choose to run profile, masking, or tokenization jobs on.

Masking Algorithms

The following terminology is around the different Algorithms that users may use to secure their data.

Term	Definition
Secure Lookup	The most commonly used type of algorithm. It is easy to generate and works with different languages. When this algorithm replaces real, sensitive data with fictional data, it is possible that it will create repeating data patterns, known as “collisions.” For example, the name “Tom” and “Peter” could both be masked as “Matt.” Because names and addresses naturally recur in real data, this mimics an actual data set. However, if you want the masking engine to mask all data to unique outputs, you should use segmented mapping, described below.
Segment Mapping	Produces no overlap or repetition in the masked data. You can mask up to a maximum of 36 values using segmented mapping. You might use this method if you need columns with unique values, such as Social Security Numbers, primary key columns, or foreign key columns. You can set the algorithm to produce alphanumeric results (letters and numbers) or only numbers.
	Allows you to state what value will replace the original data. There will be no collisions in the masked data, because it always matches the same input to the same output. For example “David” will always become “Ragu” and “Melissa” will always become “Jasmine.” The

Mapping	algorithm checks whether an input has already been mapped; if so, the algorithm changes the data to its designated output. You can use a mapping algorithm on any set of values, of any length, but you must know how many values you plan to mask. NOTE: When you use a mapping algorithm, you cannot mask more than one table at a time. You must mask tables serially.
Binary Lookup	Replaces objects that appear in object columns. For example, if a bank has an object column that stores images of checks, you can use binary lookup algorithm to mask those images. The Delphix Engine cannot change data within images themselves, such as the name on X-rays or driver's licenses. However, you can replace all such images with a new, fictional image. This fictional image is provided by the owner of the original data.
Tokenization	The only type of algorithm that allows you to reverse its masking. For example, you can use a tokenization algorithm to mask data before you send it to an external vendor for analysis. The vendor can then identify accounts that need attention without having any access to the original, sensitive data. Once you have the vendor's feedback, you can reverse the masking and take action on the appropriate accounts. Like mapping, a tokenization algorithm creates a unique token for each input such as "David" or "Melissa." The Delphix Engine stores both the token and original so that you can reverse masking later.
Min Max	Values that are extremely high or low in certain categories allow viewers to infer someone's identity, even if their name has been masked. For example, a salary of $1 suggests a company's CEO, and some age ranges suggest higher insurance risk. You can use a min max algorithm to move all values of this kind into the midrange.
Data Cleaning	Does not perform any masking. Instead, it standardizes varied spellings, misspellings, and abbreviation for the same name. For example, "Ariz," "Az," and "Arizona" can all be cleaned to "AZ."
	Helps you remove sensitive data that appears in free-text columns such as "Notes." This type of algorithm requires some expertise to use, because you must set it to recognize sensitive data within a block of text.

Free Text Redaction	One challenge is that individual words might not be sensitive on their own, but together they may be. This algorithm uses profiler sets to determine which information it needs to mask. You can decide which expressions the algorithm uses to search for material such as addresses. For example, you can set the algorithm to look for “St,” “Cir,” “Blvd,” and other words that suggest an address. You can also use pattern matching to identify potential sensitive information. For example, a number that takes the form 123-45-6789 is likely to be a Social Security Number. You can use free text redaction algorithm to show or hide information by displaying either a “black list” or a “white list.”

Profile Job Concepts

The following set of concepts are options available to the user for configuring a profiling job.

Term	Definition
Job Name	A free-form name for the job you are creating. Must be unique.
Multi-Tenant	Check the box if the job is for a multi-tenant database. This option allows existing rulesets to be-reused to mask identical schemas via different connectors. The connector can be selected at job execution time.
Rule Set	Select a ruleset that this job will execute against.
No. of Streams	The number of parallel streams to use when running the jobs. For example, you can select two streams to run two tables in the ruleset concurrently in the job instead of one table at a time.
Min Memory (MB) *optional*	Minimum amount of memory to allocate for the job, in megabytes.
Max Memory (MB)	Maximum amount of memory to allocate for the job, in megabytes.

optional	
Feedback Size *optional*	The number of rows to process before writing a message to the log. Set this parameter to the appropriate level of detail required for monitoring your job. For example, if you set this number significantly higher than the actual number of rows in a job, the progress for that job will only show 0 or 100%
Profile Sets *optional*	The name of a profile set, which is a subset of expressions (for example, a subset of financial expressions).
Comments *optional*	Add comments related to this job.
Email *optional*	Add email address(es) to which to send status messages. Separate addresses with a comma (,).

Masking Job Concepts

These concepts are options available to the user for configuring a masking job.

Term	Definition
Job Name	A free-form name for the job you are creating. Must be unique across the entire application.
Masking Method	Select either In-Place or On-The-Fly.
Multi-Tenant	Check box if the job is for a multi-tenant database.
Rule Set	Select a ruleset for this job to execute against.
Masking Method	Select either In-place or On-the-fly.
Min Memory (MB) optional	Minimum amount of memory to allocate for the job, in megabytes.

Max Memory (MB) optional	Maximum amount of memory to allocate for the job, in megabytes.
Update Threads	The number of update threads to run in parallel to update the target database. For database using T-SQL, multiple update/insert threads can cause deadlock. If you see this type of error, reduce the number of threads that you specify in this box.
Commit Size	The number of rows to process before issuing a commit to the database.
Feedback Size	The number of rows to process before writing a message to the logs. Set this parameter to the appropriate level of detail required for monitoring your job. For example, if you set this number significantly higher than the actual number of rows in a job, the progress that job will show 0% or 100%.
Bulk Data *optional*	For In-Place masking only. The default is for this check box to be clear. If you are masking very large tables in-place and require performance improvements, check this box. Delphix will mask data to a flat file, and then use inserts instead of updates to bulk load the target table.
Disable Trigger *optional*	Whether to automatically disable database triggers. The default is for this check box to be clear and therefore not perform automatic disabling of triggers.
Drop Index *optional*	Whether to automatically drop indexes on columns which are being masked and automatically re-create the index when the masking job is completed. The default is for this check box to be clear and therefore not perform automatic dropping of indexes.
Prescript *optional*	Specify the full pathname of a file that contains SQL statements to run before the job starts, or click Browse to specify a file. If you are editing the job and a pre script file is already specified, you can click the Delete button to remove the file. (The Delete button only appears if a prescript file was already specified.) For information about creating your own prescript files, see Create SQL Statements to Run Before and After Jobs.

Postscript *optional*	Specify the full pathname of a file that contains SQL statements to be run after the job finishes, or click Browse to specify a file. If you are editing the job and a postscript file is already specified, you can click the Delete button to remove the file. (The Delete button only appears if a postscript file was already specified.) For information about creating your own postscript file, see Creating SQL Statement to Run Before and After Jobs.
Comments *optional*	Add comments related to this masking job.
Email *optional*	Add email address(es) to which to send status messages.

Kerberos Configuration

Introduction

As of 5.3.0.0, the Delphix Masking Engine supports Kerberos authentication for Oracle, MS SQL Server, and Sybase connections. Utilizing this service requires the presence of a Kerberos Key Distribution Center (KDC) server as well as additional configuration actions to be done on both the the Masking Engine and the database. This document presents configuration instructions for enabling and using Kerberos on the Delphix Masking Engine, as well as reference configurations for enabling Kerberos on the Databases. Although other configurations are possible, the configurations in this document have been validated by Delphix.

Terminology

Throughout this document, the following example values are used. To recreate these reference environments, these values must be replaced with real values appropriate for your network environment:

- .bar.com - the DNS domain of then network
- BAR.COM - the Kerberos domain
- me-host - the hostname of the masking engine
- foo-kcd - the hostname KDC server
- krbuser - the kerberos principal to be granted access to the database for masking

Configuring Kerberos on the Appliance

This section details the steps required to configure Kerberos on your appliance.

Step 1 On the Delphix System Setup CLI, enable the Kerberos feature.

Note
You may see a warning indicating that special permission is required to enable Kerberos. This warning can be ignored when enabling Kerberos for use with Masking only.
In the following examples, `me-hosts` is the hostname of your masking engine.

```
$ ssh sysadmin@me-host.bar.com me-host> system me-host system> enableFeatureFlag me-host
```

```
system enableFeatureFlag *> set name=KERBEROS me-host system enableFeatureFlag *> commit me-host system> exit
```

Step 2 On the Delphix System Setup CLI, configure and enable Kerberos.

```
$ ssh sysadmin@me-host.bar.com me-host service> kerberos me-host service kerberos> update me-host service kerberos update *> set name=Kerberos_Conf me-host service kerberos update *> edit kdcs me-host service kerberos update kdcs *> edit 0 me-host service kerberos update kdcs 0 *> set hostname=foo-kcd.bar.com me-host service kerberos update kdcs *> back me-host service kerberos update *> set realm=BAR.COM me-host service kerberos update *> set principal=krbuser me-host service kerberos update *> set keytab=_krbuser_keytab_base64_ me-host service kerberos update *> commit
```

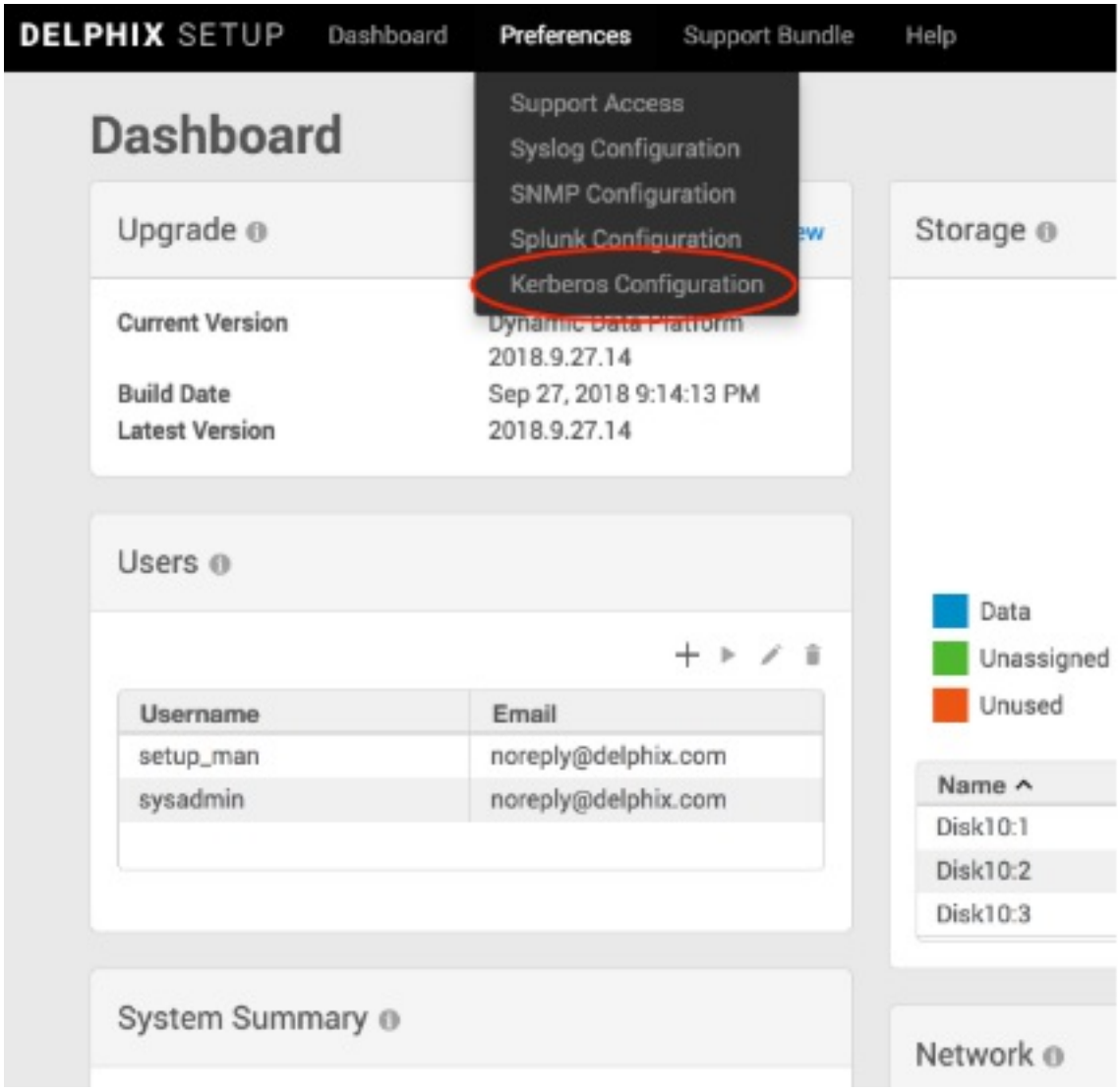
In this case, *krbuser_keytab_base64* is the base64 encoded contents of the keytab file for krbuser. The kerberos keytab for a user is typically available from your kerberos administrator.

To display a keytab file in base64 encoding use:

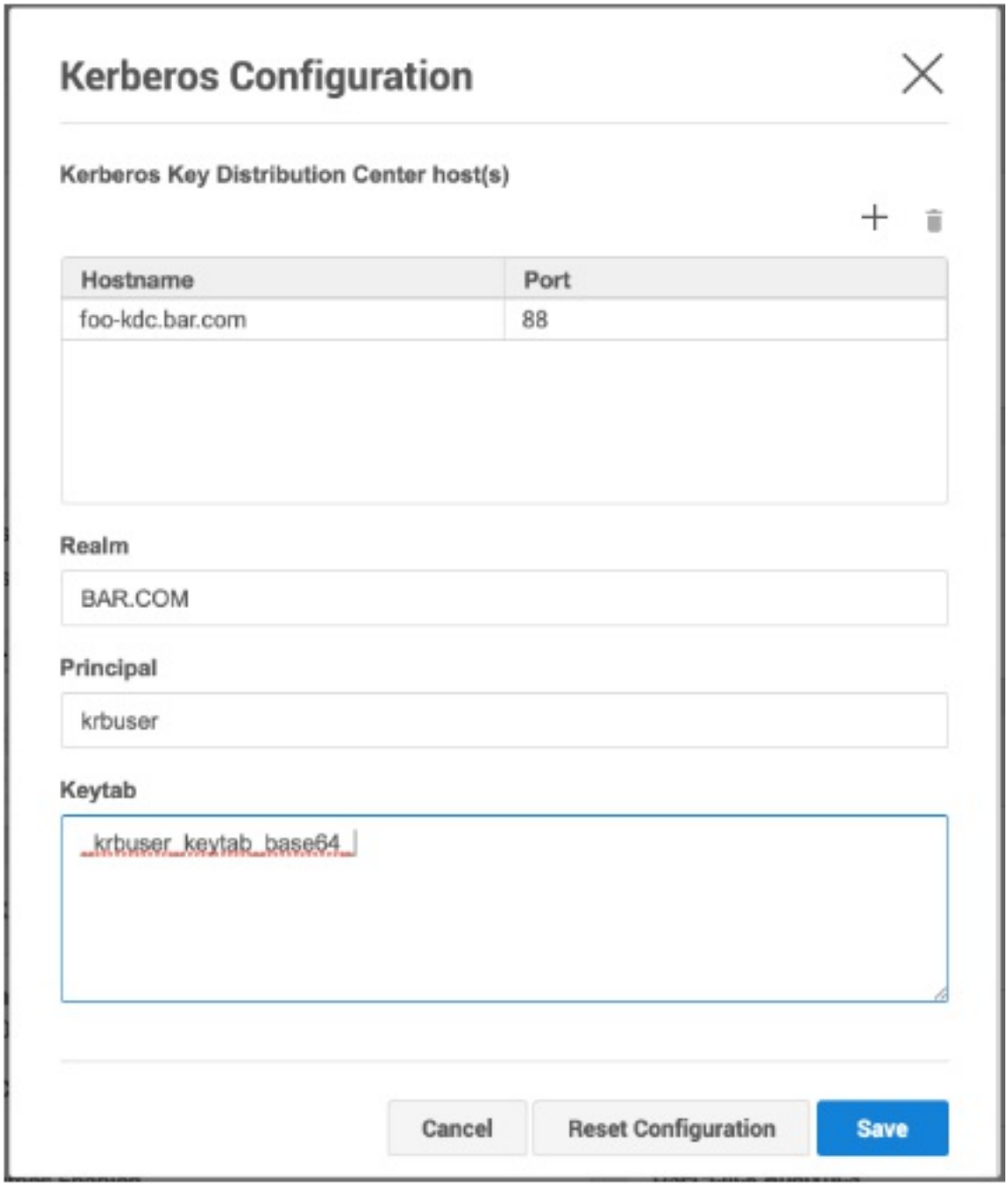
```
$ base64 ~/krbuser.keytab
```

Step 2 Alternatively - On the Delphix Server Setup UI configure and enable Kerberos:

a. From the Preferences menu select Kerberos Configuration.

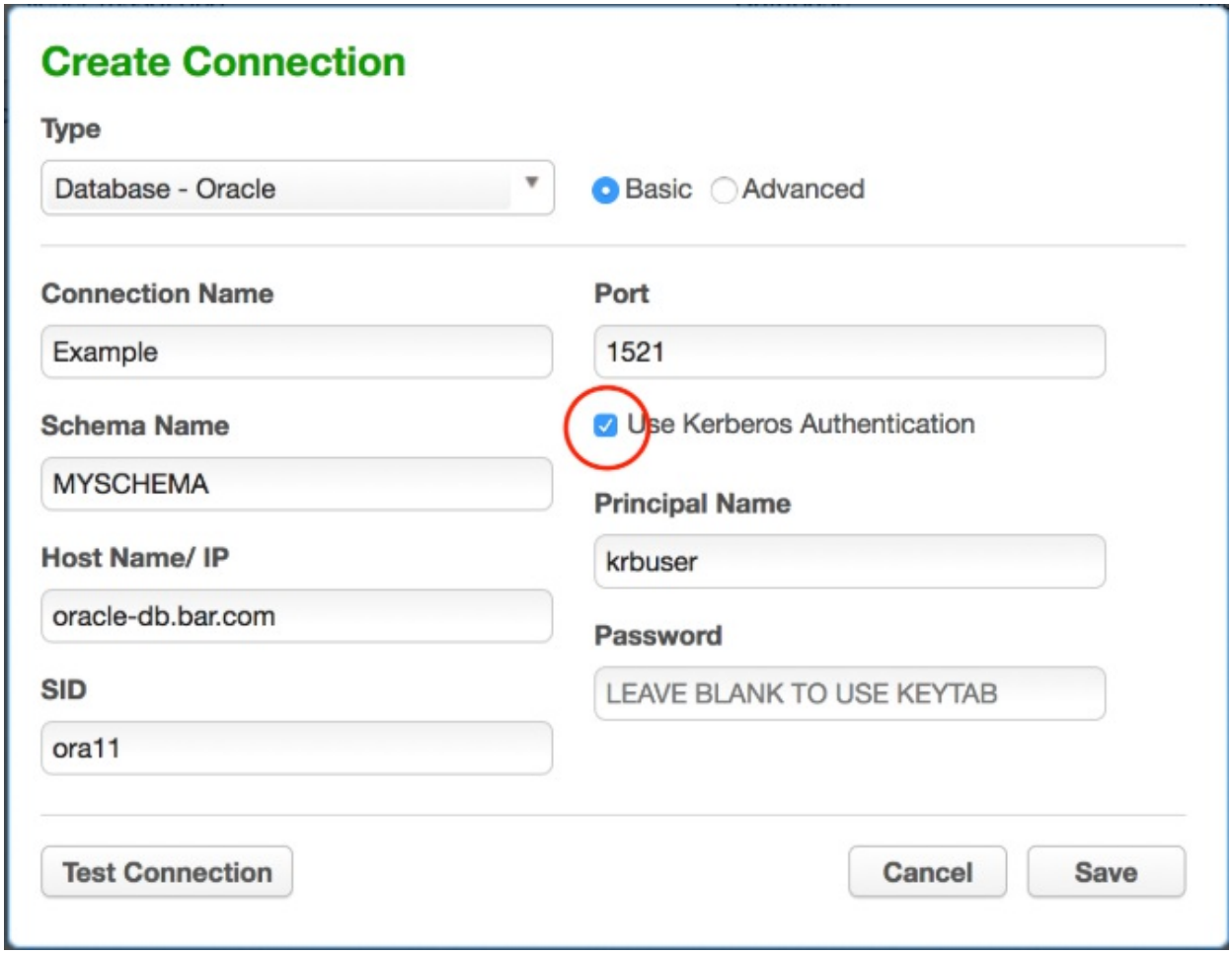


b. Add record(s) for your KDCs, and populate other fields appropriately for your network environment. Upon pressing Save, your configuration will be tested. If the engine is able to authenticate to the KDC with the supplied configuration, the configuration is applied immediately.



Creating Maskings Database Connectors using Kerberos

Once the Delphix Appliance is configured for Kerberos, creating Connectors using Kerberos authentication is simple:



Assuming you are using the same user principal configured in Server Setup, the keytab will be used and it is unnecessary to enter a password in the Connector definition.

For Sybase database Connectors, it is necessary to supply the service principal name as an additional configuration item. For Oracle DB, this value is determined automatically. For MS SQL Server it is determined based on the reverse DNS mapping of the Server Name (refer to the section on MS SQL Server below).

Reference Database Configurations

The following are a series of reference kerberos configuration procedures and troubleshooting notes for the supported databases. These are meant to serve as examples to be further customized according to the user's specific network environment and security needs.

Oracle Database

Overview

This document describes how to set up an Oracle DB instance for kerberized connections. The following steps are described:

- Creating a service principal and adding it to the DB system
- Configuring the database to use kerberos authentication
- Creating DB users identified via kerberos
- Troubleshooting tips

Prerequisites

This document assumes you already have a kerberized network environment with an MIT Kerberos KDC. These procedures have been tested successfully with Oracle database versions 11.2.0.2, 11.2.0.4 and 12.2.1. Oracle database version 12.1.0.1 did not work in our testing.

You will need the following from your kerberos environment:

- The krb5.conf file
- A user principal and associated password or keytab you'd like to use to log into the database
- The ability to create a service principal for the Oracle DB and retrieve the associated keytab

This section of the document uses these example values in addition to those mentioned above:

- The oracle database is: ora-db.bar.com.
- The oracle service name is: oracle

Creating the Oracle Service Principal

The service principal will be named:
/@
Given our default values above, this works out to:
oracle/ora-db@bar.com

Notice that the hostname is whatever the database system thinks its hostname is - that is, the output of "uname -n" on the database system, rather than the actual DNS name of the database system. Typically, these values would be the same, but this is not always the case.

On the KDC, run:

```
# kadmin.local kadmin.local: addprinc -randkey oracle/ora-db@bar.com kadmin.local: ktadd -norandkey -k /var/tmp/ora-db.keytab oracle/ora-db@bar.com
```

Copy the resulting keytab file (/var/tmp/ora-db.keytab) to the Oracle DB system at this location: /etc/v5srvtab

As root on the Oracle DB system, ensure that the keytab has the correct permissions:

```
# chown root:oinstall /etc/v5srvtab
```

```
# chmod 440 /etc/v5srvtab
```

Finally, this is a good opportunity to copy /etc/krb5.conf from the KDC to /etc/krb5.conf on the Oracle DB system. This file should be readable by all users.

Configuring the Oracle Database for Kerberos

Log into the Oracle DB system as the appropriate use for the database in question.

```
$ cd $ORACLE_HOME $ vi network/admin/sqlnet.ora
```

Add the following for Oracle 11:

```
SQLNET.KERBEROS5_CONF=/etc/krb5.conf SQLNET.AUTHENTICATION_SERVICES=(BEQ,KERBEROS5) SQLNET.KERBEROS5_CONF_MIT=true SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=oracle
```

Or the following for Oracle 12:

```
NAMES.DIRECTORY_PATH=(TNSNAMES, EZCONNECT, HOSTNAME) SQLNET.KERBEROS5_CONF=/etc/krb5.conf SQLNET.AUTHENTICATION_SERVICES=(BEQ,KERBEROS5PRE,KERBEROS5) SQLNET.KERBEROS5_CONF_MIT=true SQLNET.AUTHENTICATION_KERBEROS5_SERVICE=oracle
```

If the database is Oracle 11 (not necessary on Oracle 12):
`$ vi dbs/init.ora`
Add this line at the end: `OS_AUTHENT_PREFIX=""`

Creating a DB User Identified via Kerberos

Log into the Oracle DB system as the appropriate database user and open a database session as the DBA:

```
$ sqlplus / as sysdba
```

On Oracle 12, you may wish to alter your session to create the user in one of the PDBs:

```
SQL> alter session set container=MYPDB;
```

Create the user that will connect to the DB using kerberos:

```
SQL> create user krbdbuser identified externally as 'krbuser@BAR.COM';
```

Grant the user privileges necessary for masking.

This example grants all privileges for the sake of simplicity:

Oracle 11:

```
SQL> grant all privilege to krbdbuser;
```

Oracle 12: (Customize permissions as necessary for your environment).

`SQL> grant connect,resource to krbdbuser;

SQL> grant create tablespace, drop tablespace to krbdbuser;

SQL> grant create table to krbdbuser;

SQL> grant create sequence to krbdbuser;

SQL> grant select_catalog_role to krbdbuser;

SQL> grant unlimited tablespace to krbdbuser;

SQL> grant select_catalog_role to krbdbuser;

SQL> grant alter system to krbdbuser;

SQL> grant sysoper to krbdbuser;

SQL> grant dba to krbdbuser;`

Troubleshooting Tips

- Connecting via JDBC with kerberos authentication from Delphix Masking involves two steps - a kerberos login, followed by JDBC connect. A failure stack with an error in the

login function indicates a misconfiguration on either the engine or KDC - the engine hasn't even attempted to communicate with the database at that point. Failure stacks are saved in the debugging log for masking.
- Login exceptions that mention a checksum error mean either the password or keytab supplied doesn't match the expected password/key on the KDC for the principal you're trying to use. Server Setup verifies that your keytab works at configuration time, but it could stop working if the key for your principal is updated on the KDC.
- Prior to version 12, Oracle databases instances assume they can create/write a particular temporary file to store kerberos credentials for the DB. This means if you attempt to run multiple kerberized instances of Oracle 11 on the same system or VM, and the databases run as different system users, the first Oracle instance that performs kerberos auth will create and own this file. Kerberos authentication will fail to function on all other instances.

MS SQL Server

Overview

This is an overview of the step necessary to get your masking engine talking to a MS SQL Server database using kerberos authentication. Since Active Directory already uses Kerberos for authentication, little or no additional configuration is need on the MS SQL Database server.

The following steps are described in this section:

- Create the necessary SPNs (Service Principal Names) for your MSSQL Database service in AD
- Create the DB Connector on the masking engine
- Creating a keytab for an AD User
- Troubleshooting tips

Prerequisites

Configuring cross-realm trust between Active Directory and an MIT KDC Server is a complex topic, and will not be described here. In the absence of such a setup, it is possible to make the Delphix Appliance a kerberos client of the Active Directory (AD) Server. In this configuration, no additional KDC in necessary. The example below assumes this kind of configuration.

This section of the document uses these example values in addition to or instead of those mentioned above:

- The MSSQL server database is named mssql-db.bar.com.

- The AD user configured for masking access to the MSSQL database is aduser (rather than krbuser in other examples elsewhere in this document).
- The AD user that start the MS SQL Server service on the DB Server is dbuser.

Creating SPNs for the Database Service

MS SQL Server service will typically register several SPNs with AD upon startup. However, there are several conditions which can cause these SPNs to not be registered successfully, or to be registered with service names other than those that are expected by the jTDS JDBC driver employed by Delphix Masking.

The service principal name for an MS SQL Server expected by Delphix Masking is:
MSSQLSvc/:
For example, the SPN for our example MS SQL Server would be:
`MSSQLSvc/mssql-db.bar.com:1433`

In addition, it is required that a reverse mapping exist in DNS from the IP address of the MS SQL Server system to the FQDN registered.

The following commands may be run in powershell on the MS SQL Server to assist in debugging SPN related issues:

List all SPNs for dbuser:

```
setspn -L -U dbuser
```

Deleting an old SPN associated with dbuser:

```
setspn -U -D MSSQLSvc/other-server.ad.bar.com:SQL2008R2 dbuser
```

Here's how to create the SPN describe above:

```
setspn -U -S MSSQLSvc/mssql-db.bar.com:1433 dbuser
```

Creating the Database Connector on the Masking Engine

Once the above steps are complete, creating the database connector can be performed using the procedure above. Enter the username and optionally, password of the AD user in the Connector definition. Be sure that the AD user has the sufficient access to the MS SQL Database for masking.

The password field can be left blank when creating the connector if the user is the same user configured in Server Setup for the appliance. Since keytabs

are not typically used in an AD environment, it may be useful to create one manually, to avoid having a password in the DB Connector.

Creating a keytab file for an AD user

On a unix or MAC system with MIT kerberos CLI utilities installed:

```
# ktutil

ktutil: addent -password -p krbuser -k 1 -e arcfour-hmac

<type password for krbuser>

ktutil: addent -password -p krbuser -k 1 -e aes128-cts-hmac-sha1-96

<type password for krbuser>

ktutil: addent -password -p krbuser -k 1 -e aes256-cts-hmac-sha1-96

<type password for krbuser>

ktutil: write_kt /var/tmp/krbuser.keytab

ktutil: exit

# base64 /var/tmp/krbuser.keytab ;# This is string to user for keytab in Server Setup kerberos configuration
```

Note
kvno doesn't matter when using kerberos keytabs with AD. The password must match the active password for the AD user in question.

Troubleshooting Tips

The client uses the incorrect service name
This will typically manifest an exception mentioning cred, like:

```
Caused by: org.ietf.jgss.GSSException: No valid credentials provided (Mechanism level: Fail to create credential. (63) - No service creds)

at sun.security.jgss.krb5.Krb5Context.initSecContext(Krb5Context.java:770)

at sun.security.jgss.GSSContextImpl.initSecContext(GSSContextImpl.java:248)
```

`at sun.security.jgss.GSSContextImpl.initSecContext(GSSContextImpl.java:179)`

`at com.microsoft.sqlserver.jdbc.KerbAuthentication.intAuthHandShake(KerbAuthentication.java:163) ... 101 common frames omitted`

`Caused by: sun.security.krb5.internal.KrbApErrException: Fail to create credential. (63) - No service creds`
`at sun.security.krb5.internal.CredentialsUtil.acquireServiceCreds(CredentialsUtil.java:162)`

`at sun.security.krb5.Credentials.acquireServiceCreds(Credentials.java:458)`

`at sun.security.jgss.krb5.Krb5Context.initSecContext(Krb5Context.java:693) ... 104 common frames omitted`

Why might this happen:

- You're using the JTDS JDBC driver, and your MSSQL Server's IP address doesn't have a reverse mapping in DNS. In this case, the driver may construct a service name like: MSSQLSvc/: and try to use that. Either correct DNS to have a valid reverse mapping for the IP of your SQL server, or manually add an SPN to active directory for the name the JDBC client is trying to use:
  - Determine the user that starts MSSQL Server on your DB machine.
  - From powershell, do: setspn -AU MSSQLSvc/:1433
    Example: setspn -AU MSSQLSvc/10.43.100.101:1433 AD\dbuser
- The database server has multiple DNS names (FQDNs). In this case, SPNs may be registered only for some of them. It may be necessary to add SPNs for the other FQDNs as above.
- The MS SQL Server didn't automatically register an SPN. There is a limit (in the thousands) to the number of SPNs that may be registered for a given AD user. It is quite possible to hit this limit in an environment where many MS SQL Server VMs are actively created and destroyed with the same configuration.

Note
In Active Directory, setspn isn't creating a service principal with distinct key as is typical for services on MIT KDCs - rather it's mapping the service principal to the key for the AD user in question.

The SPN for the SQL Server is registered to the incorrect AD account

Manifests as an exception with this text: GSS failure: Defective token detected (Mechanism level: AP_REP token id does not match!)

Resolution: From powershell on the MS SQL Server:

```
PS> setspn -Q <SPN>
```

This will show what user has the SPN registered.

```
PS> setspn -U -D <SPN> <WRONG_ACCT>
```

This will unregister the SPN from that user

```
PS> setspn -AU <SPN> <CORRECT_ACCT>
```

Sybase

Creating a principal and corresponding keytab on the KDC

1. SSH into the KDC as the user with sufficient privileges to run kadmin.local
2. Run the kerberos configuration CLI with kadmin.local
3. Add a new principal you want to authenticate as later with:
   `add_principal <principalName>`

   We're going to continue to use krbuser as our example kerberos principal.
4. Once you've created the principal and provided it a password, we need to generate a keytab for it. Do so via the following command:

   `ktadd -norandkey -k v5srvtab krbuser`

   In this case, v5srvtab is the keytab filename, and it will be placed into whatever directory you've invoked kadmin.local from. Presumably this will be the home directory of the machine.
5. You now have everything you need done on the KDC, but you will need your keytab file later as well as the krb5.conf file that is located in the home directory of the KDC, so consider moving them somewhere (probably your local machine) that will be convenient for you to access later.

Configuring the Sybase image for Kerberos

1. Start up a Sybase database.

- Note: Each sybase database machine may have multiple sybase instances running on it at a given point in time. In this case, I am configuring the ASE_1550_S5 instance, but these steps can be done on any instance so long as you change the $SYBASE_HOME directories accordingly.

2. Connect to the particular sybase instance you are working on and invoke the following sql statement:

   ```
   sp_configure 'use security services', 1
   ```

3. Continue to create a user with the same name as the principal name you created previously on the KDC, in this case krbuser:

   ```
   sp_addlogin krbuser, <password>
   ```

4. Change your $SYBASE environment variable to point to the sybase directory for whichever instance you are configuring. In this case, we want to do:

   ```
   export SYBASE=/opt/sybase/15-5
   ```

5. Open the $SYBASE/interfaces file, and find the header for whichever Sybase instance you are configuring. In our case, it is ASE_1550_S5. You should see something that looks like this:

   ```
   ASE1550_S5
   ```

   ```
   master tcp ether 10.43.89.241 5500
   ```

   ```
   master tcp ether localhost 5500
   ```

   ```
   query tcp ether 10.43.89.241 5500
   ```

   ```
   query tcp ether localhost 5500
   ```

   You want to add the following line to this:

   ```
   secmech 1.3.6.1.4.1.897.4.6.6
   ```

   This line is static, while the other lines in this section are dynamically generated for your instance. So, your final result should look something like this:

   ```
   ASE1550_S5
   ```

`master tcp ether 10.43.89.241 5500` < your numbers will vary

`master tcp ether localhost 5500` < your numbers will vary

`query tcp ether 10.43.89.241 5500` < your numbers will vary

`query tcp ether localhost 5500` < your numbers will vary

6. Navigate to $SYBASE/OCS-15_0/config. You should see libtcl64.cfg and libtcl.cfg

   a. Change the contents of libtcl64.cfg to be this:

```
[DIRECTORY]

;ldap=libsybdldap.so ldap://ldaphost/dc=sybase,dc=com

[SECURITY]
csfkrb5=libsybskrb64.so secbase=@bar.com libgss=/lib64/libgssapi_krb5.so.2.2
[FILTERS]
;ssl=libsybfssl.so
```

   b. Change the contents of libtcl.cfg to be this:

```
[DIRECTORY]

;ldap=libsybdldap.so ldap://ldaphost/dc=sybase,dc=com

[SECURITY]

csfkrb5=libsybskrb.so secbase=@bar.com libgss=/lib64/libgssapi_krb5.so.2.2

[FILTERS]

;ssl=libsybfssl.so
```

   c. Note that the @bar.com value is our realm name that is determined by the KDC. Realistically, you should never have to deal with this, and it should never change, but if for some reason it does, that value needs to be updated.

7. Create a directory for those Kerberos config files you created on the KDC in the previous set of steps:

```
sudo mkdir /krb
```

Copy into /krb your keytab file v5srvtab and config file krb5.conf that you took off of the KDC earlier.

8. Head to $SYBASE/ASE-15_0/install and open the RUN_ASE1550_S5 file. We're going to add information so that Sybase knows where to find our keytab and our krb5.conf file, so change the content to look like this:

```
#!/bin/sh

#

# ASE page size (KB) : 4096

# Master device path: /opt/sybase/devices/data5/S5_master.dat

# Error log path: /opt/sybase/errorlogs/ASE1550_S5.log

# Configuration file path: /opt/sybase/15-5/ASE-15_0/ASE1550_S5.cfg

# Directory for shared memory files: /opt/sybase/15-5/ASE-15_0

# Adaptive Server name: ASE1550_S5

#

export **KRB5_KTNAME**=/krb/v5srvtab

export **KRB5_CONFIG**=/krb/krb5.conf

/opt/sybase/15-5/ASE-15_0/bin/dataserver \

-kASE1550_S5@bar.com \

-d/opt/sybase/devices/data5/S5_master.dat \

-e/opt/sybase/errorlogs/ASE1550_S5.log \

-c/opt/sybase/15-5/ASE-15_0/ASE1550_S5.cfg \

-M/opt/sybase/15-5/ASE-15_0 \

-sASE1550_S5 \
```

9. Reboot the Sybase instance you're working so that it reads in all of these config changes.

10. Connect to the Sybase instance as the dbo user so that you may give dbo privileges to your kerberos authentication login on a particular database within the instance. Below is an example of doing so with the database potatoes:

```
>> sql5
```

```
1> use potatoes
```

```
2> go
```

```
1> sp_addalias instructions, dbo
```

```
2> go
```

```
Alias user added.
```

```
(return status = 0)
```

11. Now, to access the Sybase instance via kerberos and confirm success, you can do the following set of commands (I put these three lines into a script called connect.sh for future convenience):

```
#!/bin/sh
```

```
kinit -k -t /krb/v5srvtab <yourPrincipalName>
```

```
export SYBASE='/opt/sybase/15-5'
```

```
/opt/sybase/15-5/OCS-15_0/bin/isql64 -V -SASE1550_S5
```

Testing by creating a Kerberos Connector on the Delphix Engine

1. Start by configuring your engine for kerberos. SSH into the engine as the delphix user and run the following command:

```
/opt/delphix/server/bin/jmxtool tunable set enabled_features KERBEROS true
```

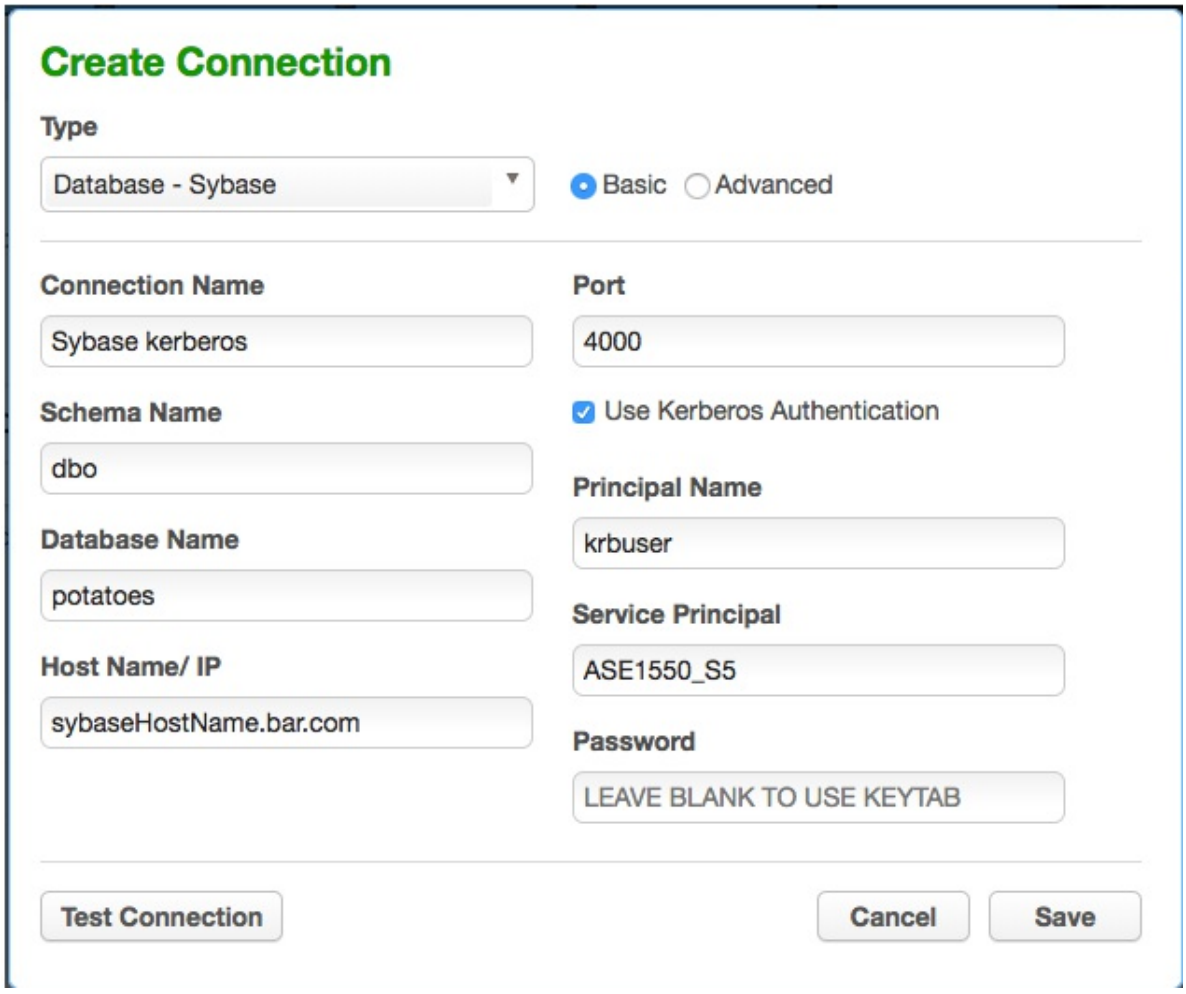
2. Log into the virtualization engine and proceed through first-time setup if you need to.

3. Once first-time setup is complete, log into the Delphix Setup page, proceed to Preferences > Kerberos Configuration. Add the information for your KDC to configure it with the principal name you created earlier, krbuser. You can get the keytab by running the following command on your keytab file:

   base64 v5srvtab

   Copy the output as plaintext into the keytab field of the kerberos configuration box.

   Finally, create a Sybase connector with parameters that look like this, and if your “test connection” attempt succeeds you’re all set!



Audit Logs

Delphix helps you keep a record of user actions taken in the UI or directly through our REST APIs. You can access these audit logs directly from our UI or through our APIs.

Audit Log UI Page

The Audit Log page can be found in the UI under the Audit tab. This page contains information on what action occurred, the user that performed the action and the time at which the action occurred. It also provides the ability to filter based on:

- user
- time range
- arbitrary search string
- action type or action target, or both (create, connector or create database connector)

Audit Log APIs

With 5.3.2.0 Delphix introduced an endpoint to get all Audit Logs. This endpoint contains the user name, action type, target, status, start time, and end time.
For more information please refer to API documentation.

What Gets Logged?

User actions are categorized into the following:

Cancel	Create	Delete	Edit	Export	Get	Get All
Import	Lock	Login	Logout	Run	Test	Unlock

The objects that user actions target are categorized into the following:

Algorithm	Analytics	Application	Application Log	Application Setting	Asyn Task
Column Metadata	Database Connector	Ruleset Connector	Database Ruleset	Domain	Encrypt Key

Execution	File Connector	File Download	File Field Metadata	File Format	File Metada
File Upload	LDAP	Mainframe Dataset Connector	Mainframe Dataset Field Metadata	Mainframe Dataset Format	Mainfra Dataset Metada
Masking Job	Profile Expression	Profile Job	Profile Set	Re Identification Job	Role
SSO	Syncable Object	System Information	Table Metadata	Tokenization	User

Retention Policy

The default policy stores the last one million Audit Log entries. Any entries older than the most recent million are removed daily. Additionally, there is a fail-safe mechanism that prevents an attacker from forcing an unbounded number of actions to be logged to overload the system's disk space. In the event that such an attack occurs, Delphix also logs it to the application logs.

Recommendation

If a full record of all Audit Log entries is desired, Delphix recommends using the new API to periodically retrieve new entries to the Audit Logs.

Preparing Oracle Database for Profiling/Masking

Before masking your data, it is important to prepare your database. This section explains the required changes, reasons for the changes, and instructions on how to make the changes.

Archive Logging

What is Archive Logging?

Oracle Database lets you save filled groups of redo log files to one or more offline destinations, known collectively as the archived redo log, or more simply the archive log. The process of turning redo log files into archived redo log files is called archiving. This process is only possible if the database is running in ARCHIVELOG mode. You can choose automatic or manual archiving.

Why is it important to make this change?

Archive logging will slow down masking processes and absorb CPU resources that could be used by the masking process. Furthermore, since masking will change every row in every table being masked logs are only needed for short term recovery and transaction backout.

The choice of whether to enable the archiving of filled groups of redo log files depends on the availability and reliability requirements of the application running on the database. If you cannot afford to lose any data in your database in the event of a disk failure, use ARCHIVELOG mode. The archiving of filled redo log files can require you to perform extra administrative operations.

How exactly do I make this change? (exact commands, etc).

```
ALTER DATABASE NOARCHIVELOG;
```

DB/VDB Memory Allocation

What is SGA?
A system global area (SGA) is a group of shared memory structures that contain data and control information for one Oracle database instance. If multiple users

are concurrently connected to the same instance, then the data in the instance's SGA is shared among the users. Consequently, the SGA is sometimes called the shared global area.

An SGA and Oracle processes constitute an Oracle instance. Oracle automatically allocates memory for an SGA when you start an instance, and the operating system reclaims the memory when you shut down the instance. Each instance has its own SGA.

The SGA is read/write. All users connected to a multiple-process database instance can read information contained within the instance's SGA, and severa
l processes write to the SGA during execution of Oracle.
When automatic SGA memory management is enabled, the sizes of the different SGA components are flexible and can adapt to the needs of a workload without requiring any additional configuration. The database automatically distributes the available memory among the various components as required, allowing the system to maximize the use of all available SGA memory. Make sure the DB/VDB memory allocation is sufficient for the workload. Delphix's best practices for sizing a VDB will handle most masking requirements. If you plan to run many concurrent masking jobs a small memory allocation will negatively impact performance of the masking jobs.

Why is it important to make this change?

To assure that masking jobs will perform at an optimum level.

How exactly do I make this change? (exact commands, etc).
Set automatic SGA memory management to enabled. If not allowed set the SGA based on the diagnosis from the AWR report generated during a masking job. The DBA is best suited to make the appropriate tuning changes to the SGA parameters for the version of oracle being masked.

Undo Tablespace Size And Undo Retention Time:

What is tablespace?
Every Oracle Database must have a method of maintaining information that is used to roll back, or undo, changes to the database. Such information consists of records of the actions of transactions, primarily before they are committed.
These records are collectively referred to as undo.

Undo records are used to:

- Roll back transactions when a ROLLBACK statement is issued
- Recover the database
- Provide read consistency
- Analyze data as of an earlier point in time by using Oracle Flashback Query
- Recover from logical corruptions using Oracle Flashback features

When a ROLLBACK statement is issued, undo records are used to undo changes that were made to the database by the uncommitted transaction. During database recovery, undo records are used to undo any uncommitted changes applied from the redo log to the datafiles. Undo records provide read consistency by maintaining the before image of the data for users who are accessing the data at the same time that another user is changing it.

Why is it important to make this change?

The masking Engine updates or inserts masked data in batches. In the case of an insert it only requires the current transaction size for the commit of each table being masked. The default per table stream is 10k rows. However, with an update the transaction is not complete until the entire table is masked. So, the more tables and more rows and the wider (size) each row is in each table, the more undo space is needed to complete the transaction. Large tables, such as DW tables or history and Audit tables, most often need an increase to the Undo space and undo Retention time for updates. If the space or time is exceeded then the masking job may fail with an ORA-01555, Snapshot too old error.

How exactly do I make this change? (exact commands, etc).

It is highly recommended to increase the Undo space and undo Retention time when running in-place jobs on large tables. A general rule of thumb is 2 or 3 times the size of the larges table(s), or if there are multiple tables running at the same time, then all tables combined. A DBA is best suited to make the necessary UDNO Space and the UNDO Retention changes.

Redo Logs Are Optimally Sized

What is Redo Logs?

The most crucial structure for recovery operations is the redo log, which consists of two or more preallocated files that store all changes made to the database as they occur. Every instance of an Oracle Database has an associated redo log to protect the database in case of an instance failure.

Why is it important to make this change?

The most important reason to make this change is to keep performance optimal. If redo logs are too small, then the log switching will occur too often, using up valuable Oracle resource.

How exactly do I make this change? (exact commands, etc).

A DBA is best suited to make these changes appropriately.

Change PCTFREE to 40-50:

What is PCTFREE?

PCTFREE and PCTUSED are used together, but PCTFREE is critical for updates. The larger the PCTFREE value the more updates can be done.

Why is it important to make this change?

PCTFREE aids in performance increases for updating Oracle during masking. The Masking Engine does many updates at the same time in batch mode. The more that can be done without DB overhead the faster the masking jobs run.

How exactly do I make this change? (exact commands, etc).

A DBA is best suited to make these changes.

Change Primary Key To ROWID:

What is ROWID?

For each row in the database, the ROWID pseudocolumn returns the address of the row. Oracle Database rowid values contain information necessary to locate a row.

Why is it important to make this change?

This is especially important in masking for performance. IF ROWID is used then Oracle will manage the updates for the rows it tracks using ROWID. This makes updates much faster. On occasion there may be a key (PK/FK/UK) or ID column with an index that is faster, but generally ROWID is the fastest.

How exactly do I make this change? (exact commands, etc).

Add ROWID as the logical key on each table in the ruleset using the Masking Engine GUI. Also, in a script you should drop foreign keys, and if possible indices and disable triggers and recreate them after the masking job has been run for any of these types of columns being masked.

Before masking your data, it is important to prepare your database. This section explains the required changes, reasons for the change, and the instructions to make the change.

Logging

What is Simple Recovery Model?

SQL Database Simple Recovery model - Automatically reclaims log space to keep space requirements small, essentially eliminating the need to manage the transaction log space. Operations that require transaction log backups are not supported by the simple recovery model.

Why is it important to make this change?

Reducing the overhead of the transaction logging and the size of the files before checkpoints increases the masking speed significantly.

How exactly do I make this change?

Using SQL Server Management Studio open the DB properties dialog box and select the “simple recovery model” or from a SQL Query tool enter “SET RECOVERY SIMPLE.” Please see _ for more details.

DB/VDB Memory Allocation

What is min/max memory in SQL Server?

Memory is allocated at the SQL Server level, so all the DBs will share the entire load. Max memory should be close the maximum available on the server.

Why is it important to make this change?

To assure that masking jobs will perform at an optimum level.

How exactly do I make this change?

Use SQL Server Management Studio and change the max memory allocation for the server.

Primary/Foreign/DMS_ROW_ID Keys

What is a key?

A key is a unique, non-null value that identifies a row in the database.

Why is it important to make this change?

Using a PK or Foreign key is critical for fast updates. When a table does not have an identity column with an index or a PK/FK then the masking engine will alter the table to have an Identity column, DMS_ROW_ID to optimize performance.

How exactly do I make this change?

A logical key can be added to a table in the Masking Engine Ruleset for each table, if there is a specific column that would find the row to update faster than the current PK/FK.

Before masking your data, it is important to prepare the database. This section explains the required changes, reasons for the change, and instructions to make the change.

Logging Archive

What is Durability Level?
Sybase has 3 Durability levels, Full, at_shutdown and no_recovery. Databases with a durability set to no_recovery or at_shutdown—whether they are in-memory or disk-resident—are referred to as low-durability databases. Data in low durability databases survives after a commit (provided you do not restart the server).

Use create database with durability=durability_level to set a database's durability level. Adaptive Server supports full, no_recovery, and at_shutdown durability levels.

Why is it important to make this change?

We recommend using the no_recovery to minimize log size and increase performance. This should be combined with setting the sp_dboption to 'trunc log on chkpt' to true and to set the sp_dboption to 'select into/bulkcopy/pllsort' to true. It is also recommended at the table level to use DML_Logging set to minimal to reduce logging DML statements, such as updates. This is best for large tables.

How exactly do I make this change? (exact commands, etc).

Use create database with durability=durability_level to set a database's durability level. Adaptive Server supports full, no_recovery, and at_shutdown durability levels.

`create database database`

```
`on data_device = 'size of device'`

`log on log_device = 'size of device'`

`with durability = no_recovery;`
```

`sp_dboption database, 'trunc log on chkpt', true;`

```
sp_dboption database, 'select into/bulkcopy/pllsort', true;
```

```
ALTER TABLE tablename SET dml_logging = minimal;
```

What is min/max memory in SQL Server?

Determining the Amount of Memory SAP ASE Needs

The total memory SAP ASE requires to start is the sum of all memory configuration parameters plus the size of the procedure cache plus the size of the buffer cache, where the size of the procedure cache and the size of the buffer cache are expressed in round numbers rather than in percentages.
The procedure cache size and buffer cache size do not depend on the total memory you configure. You can configure the procedure cache size and buffer cache size independently. Use sp_cacheconfig to obtain information such as the total size of each cache, the number of pools for each cache, the size of each pool, and so on.

Use sp_configure to determine the total amount of memory SAP ASE is using at a given moment:

```
1> sp_configure "total logical memory"
```

Parameter Name	Default	Memory Used	Config Value	Run Value	Unit	Type
total logical memory	33792	127550	63775	63775	memory pages(2k)	read-only

The value for the Memory Used column is represented in kilobytes, while the value for the Config Value column is represented in 2K pages.

The Config Value column indicates the total logical memory SAP ASE uses while it is running. The Run Value column shows the total logical memory being consumed by the current SAP ASE configuration. Your output differs when you run this command, because no two SAP ASEs are configured exactly the same.

Determine the SAP ASE Memory Configuration

The total memory allocated during system start-up is the sum of memory required for all the configuration needs of SAP ASE. You can obtain this value from the read-only configuration parameter total logical memory
.

This value is calculated by SAP ASE. The configuration parameter max memory must be greater than or equal to total logical memory. Max memory indicates the amount of memory you will allow for SAP ASE needs.

During server start-up, by default, SAP ASE allocates memory based on the value of total logical memory. However, if the configuration parameter allocate max shared memory has been set, then the memory allocated will be based on the value of max memory. The configuration parameter allocate max shared memory enables a system administrator to allocate the maximum memory that is allowed to be used by SAP ASE, during server start-up.

The key points for memory configuration are:

- The system administrator should determine the size of shared memory available to SAP ASE and set max memory to this value.

- The configuration parameter allocate max shared memory can be turned on during start-up and runtime to allocate all the shared memory up to max memory with the least number of shared memory segments. A large number of shared memory segments has the disadvantage of some performance degradation on certain platforms. Check your operating system documentation to determine the optimal number of shared memory segments. Once a shared memory segment is allocated, it cannot be released until the server is restarted.

- The difference between max memory and total logical memory determines the amount of memory available for the procedure and statement caches, data caches, or other configuration parameters.

- The amount of memory SAP ASE allocates during start-up is determined by either total logical memory or max memory. If you set alloc max shared memory to 1, SAP ASE uses the value for max memory.

- If either total logical memory or max memory is too high:

    - SAP ASE may not start if the physical resources on your machine are not sufficient.
    - If it does start, the operating system page fault rates may rise significantly and the operating system may need to be reconfigured to compensate.

Why is it important to make this change?

To assure that masking jobs will perform at an optimum level.

Primary/Foreign/DMS_ROW_ID keys to for masking Sybase:

What is a key?

A key is a unique, non-null value that identifies a row in the database.

Why is it important to make this change?

Using a PK or Foreign key is critical for fast updates. When a table does not have an identity column with an index or a PK/FK then the masking engine will alter the table to have an Identity column, DMS_ROW_ID to optimize performance.

How exactly do I make this change? (exact commands, etc).

A logical key can be added to a table in the Masking Engine Ruleset for each table, if there is a specific column that would find the row to update faster than the current PK/FK.

Note Sybase ASE will create unavoidable log entries when a table is altered and will increase the log size significantly. If needed, run the masking jobs using the On-The-Fly method to avoid log file increases.

Creating a Masking User and Privileges:

It is highly recommended to create a database user, and possibly a role, to mask. This user should not be created in production but should be created in non-Production. The following permissions are needed:

Syntax to add user and give privileges:

```
sp_adduser mask_user;
```

```
CREATE user NEWUSER;
```

```
CREATE LOGIN mask_user WITH PASSWORD Delphix_123; --THIS MUST BE DONE IN MASTER
```

```
CREATE USER mask_user IDENTIFIED BY Delphix_123;
```

```
GRANT SELECT ON PII_V2 TO mask_user;
GRANT INSERT ON PII_V2 TO mask_user;
```

```
GRANT DELETE ON PII_V2 TO mask_user;
GRANT ALTER ON PII_V2 TO mask_user;
GRANT UPDATE ON PII_V2 TO mask_user;

GRANT ALTER ANY TABLE TO mask_user;
```

Adaptive Server requires a two-step process to add a user: sp_addlogin followed by sp_adduser.

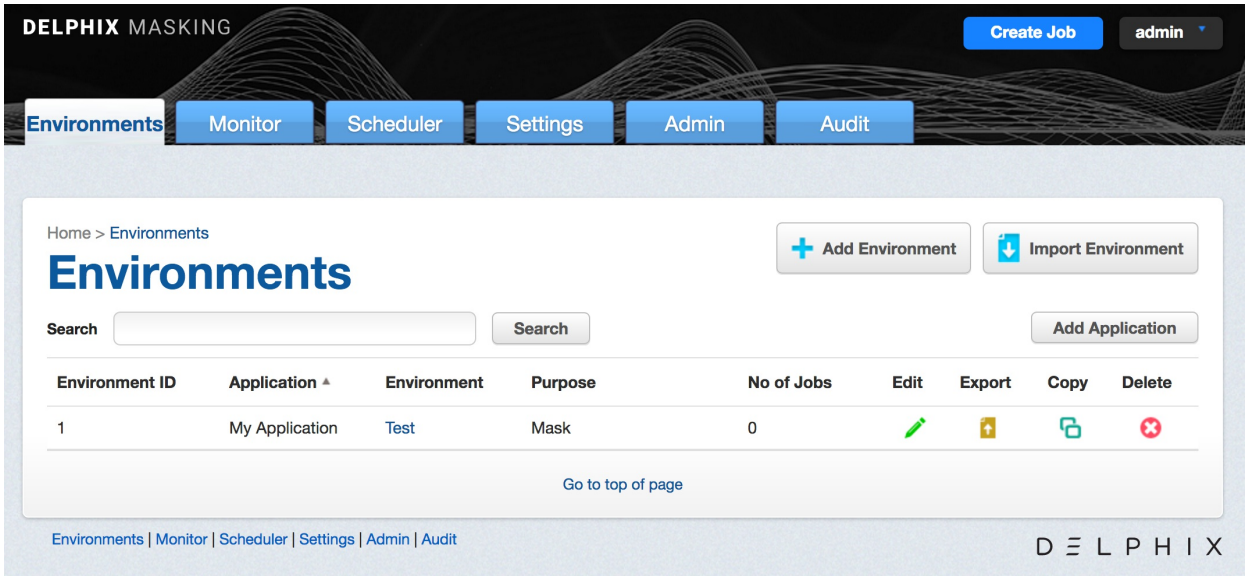
```
CREATE LOGIN MASK_SUPER_USER WITH PASSWORD Delphix_123;

sp_addlogin MASK_SUPER_USER, Delphix_123;

GRANT ROLE sa_role TO MASK_SUPER_USER;
```

Managing Environments

This section describes how you can create and manage your environments in the masking service.

As a reminder, environments are used to group certain sets of objects within the masking engine. They can be thought of as folders/containers where a specified user can create manage connectors, rulesets and jobs.

The main environment screen lists all the environments the logged in user has access to. It is the first screen that appears when a user logs in to Delphix.



The main environments screen contains the following information and actions:

- Environment ID — The numeric ID of the environment used to refer to the environment from the Masking API.
- Application — A way to indicate the name of the application whose data will be managed within this environment.
- Environment — The name of the environment.
- Purpose — The purpose of the environment.
- Jobs — The number of jobs contained within the environment.

- Edit — Edit the environment. See more details below.
- Export — Export the environment. See more details below.
- Copy — Copy the environment. See more details below.
- Delete — Delete the environment. See more details below.

The environments on the screen can be sorted by the various informational fields by clicking on the respective field. In addition, the environments listed can be filtered using the Search field. See more details below.

Adding An Application

For an environmment to be created, an application needs to be specified. Here are the steps to add an application:

1. On the main environments page, near the upper right-hand corner of the screen, click Add Application.
2. The screen prompts you for the following items:

   a. Application Name
3. Click Save to return to the Environments List/Summary screen.

Creating An Environment

Here are the steps you need to take to create an environment:

1. On the main environments page, in the upper right-hand corner of the screen, click Add Environment.
2. The screen prompts you for the following items:
   - Application Name – The name of the application to associate with the environment, for informational purposes.
   - Environment Name – The display name of the new environment.
   - Purpose – The type of masking workflow for the environment: Mask

or Tokenize/Re-Identify.

- Enable Approval Workflow – Whether or not to require approvals of inventories before masking jobs can be run in the environment.

3. Either click Save to return to the Environments List/Summary screen, or click Save & View to display the Environment Overview screen.

Editing an Environment

To change the properties of an environment, do the following

1. Click the Edit icon to the right of the environment status.
2. The popup prompts you for the following information:

   a. Environment Name

   b. Purpose

   c. Application Name

   d. Enable Approval Workflow

3. Click Save.

Exporting an Environment

For a variety of different reasons (the main one being moving environments between masking engines), you may want to export all the objects within an environment (connectors, rulesets, masking jobs, etc).

To export an environment, you have 2 different options. The first is to use Delphix's open source Masking Initializer command line tool that can be used to backup and restore a masking engine using the APIv5 endpoints. This tool is recommended when you are trying to backup/export all objects on the engine.

The second option, which will be outlined here, is to use the Export Environment option available in the Masking UI. To export an individual

environment:

1. Click the Export icon.
2. The popup fills in the following items:

   a. Environment Name

   b. File Name.
3. Click Export.

All the information for the specified environment (connectors, rule sets, inventory, jobs, and so on) is exported to an XML file.

A status popup appears. When the export operation is complete, you can click on the Download file name to access the XML file.

Importing An Environment

Once you have exported your environment, you can easily import it into another masking engine. To import an environment:

1. In the upper right-hand corner of the screen, click Import Environment.
2. The screen prompts you for the following items:

   - Application Name – The name of the application associated with this environment, for informational purposes. (An integrated test environment can have multiple applications.
   - Environment Name – The name of the environment that you want to import.
   - Purpose – The way the environment is used in the development process: Development, Gold Copy, QA, Training, and so on.
   - Enable Approval Workflow – Whether or not to require approvals of inventories before masking jobs can be run.
   - Select… – Use to browse for the XML file that contains the information you want to import. (This file must be a previously

exported Delphix Agile Data Masking environment.)

3. Either click Save to return to the Environments List/Summary screen, or click Save & View to display the
   Environment Overview screen.

Copying An Environment

A user can also easily create an exact copy of a certain environment.
This is a very powerful feature when wanting to have several similar but not exact environments but don't want to start from scratch. To copy an environment do the following:

1. Click the Copy icon to the right of the environment status.

2. The popup prompts you for the following information:

   a. Environment Name

   b. Purpose

   c. Application Name

   d. Enable Approval Workflow

3. Click Save.

Deleting An Environments

To delete an environment:

- Click the Delete icon to the right of the environment status
  and copy
  icon.

!!! warning
Clicking the Delete icon deletes EVERYTHING for that environment: connections, inventory, rule sets, and so on. It does not delete universal settings like algorithms, domains, etc.

Searching For Environments

When a large number of environments have been created on a masking engine, it may be useful to filter
the Environments List/Summary screen. To filter the environment list, do the following:

1. In the Search field in the upper left side of the screen, enter the characters to search by.

2. Click the adjacent Search button.

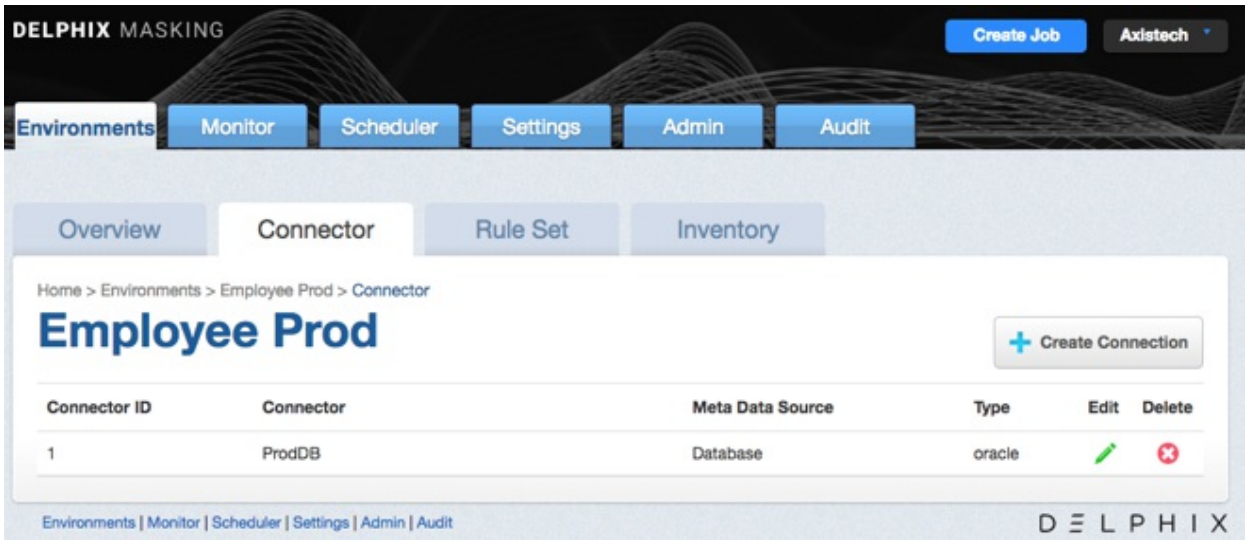
3. The screen will display only the environments whose name match the specified search characters.

To re-display the entire list of environments, clear the Search field of characters and click the Search button again.

Managing Connectors

This section describes how you can create and manage your connectors.

As a reminder, connectors are the way users define the data sources to which the masking engine should connect. Connectors are grouped within environments. In order to navigate to the connectors screen, click on an environment and then click the Connector tab.



The connectors screen contains the following information and actions:

- Connector ID — The numeric ID of the connector used to refer to the connector from the Masking API.
- Connector — The name of the connector.
- Meta Data Source — The type of connector. One of Database, File, or Mainframe.
- Type — The specific type of connector.
- Edit — Edit the connector. See more details below.
- Delete — Delete the connector. See more details below.

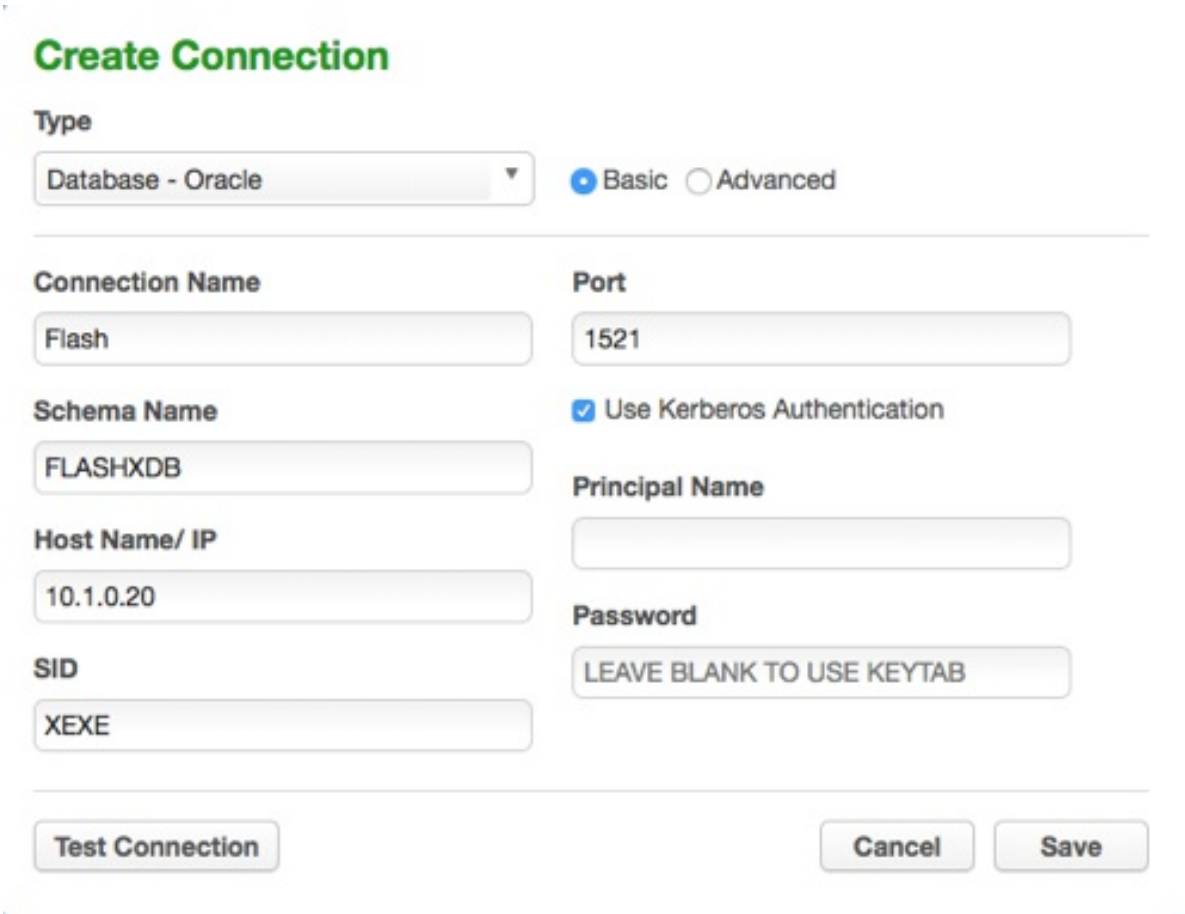
The connectors on the screen can be sorted by the various informational fields by clicking on the respective field.

Creating a Connector

To create a new connector:

1. In the upper right-hand corner of the Connector tab, click Create Connection. The Create Connection window appears, prompting you for connection information for the data source you would like to connect to. The required information will change depending on the Type of data source you select. For more details on what info is needed to connect to different types (Oracle, AWS RDS, etc) see sections below.

2. Several of our connector types offer two different modes of connecting, Basic and Advanced Mode. Advanced Mode gives you the ability to specify the exact JDBC URL & add parameters that may not be available in Basic Mode.



   The fields that appear on the Connector screen are specific to the selected Connector Type (see Connector Types below).

3. Click Save.

Editing a Connector

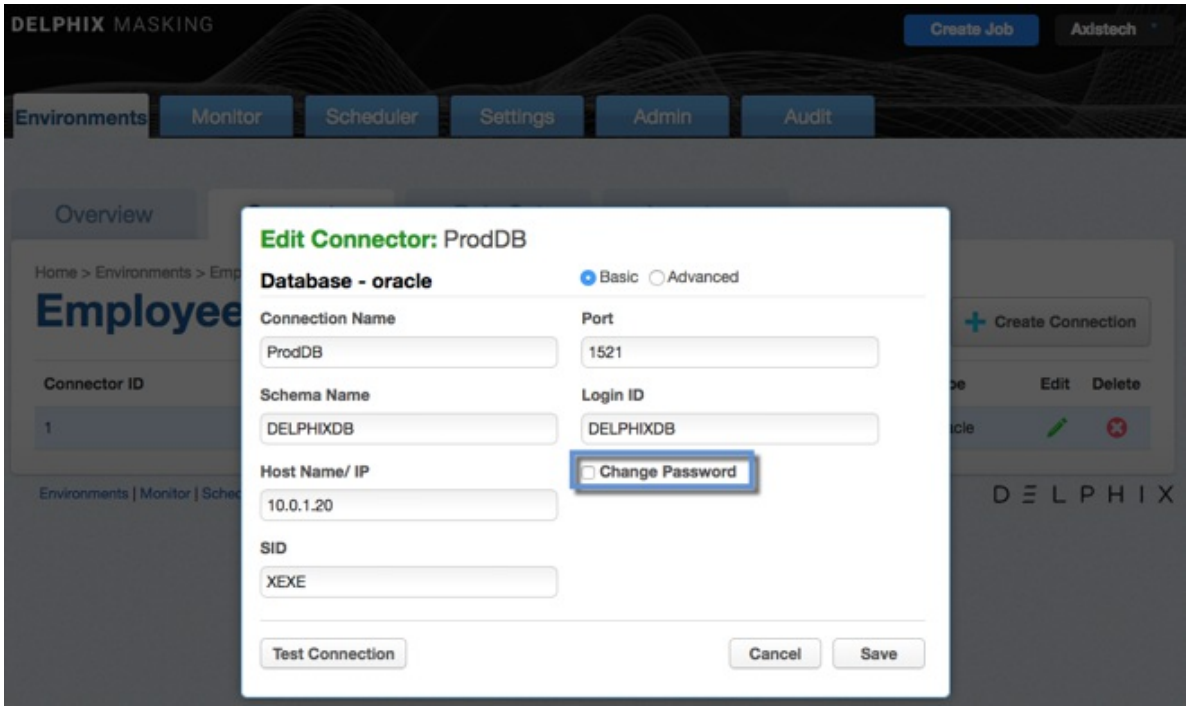
To edit a connector:

1. In the Connector tab, click the Edit icon for the connector you want to edit.

2. Change any information necessary. To change the password:

    i. Select the checkbox next to Change Password.

    ii. In the field that appears, enter the new password.



3. Click Save.

Deleting a Connector

To delete a connector, click the Delete icon to the far right of the connector name.

Warning:
When you delete a connector, you also delete its rule sets and inventory data.

Connector Types

Database Connectors

The fields that appear are specific to the DBMS Type you select. If you need assistance determining these values, please contact your database administrator.

You can only create connectors for the databases and/or files listed. If your database or file type is not listed here, you cannot create a connector for it.

- Connection Type — (Oracle, MS SQL Server, and Sybase only) Choose a connection type:
    - Basic — Basic connection information.
    - Advanced — The full JDBC connect string including any database parameters.
- Connection Name — The name of the database connector (specific for your Delphix application).
- Schema Name — The schema that contains the tables that this connector will access.
- Database Name — The name of the database to which you are connecting.
- Host Name/ IP — The network host name or IP address of the database server.
- Use Kerberos Authentication - (Oracle only, optional) Whether to use kerberos to authenticate to the database. This box is clear by default. Before Kerberos may be used, the appliance must be properly configured - refer to these instructions (link to appliance kerberos configuration instructions[1]). If this box is checked, the application authenticates with the kerberos KDC before connecting to the database, then uses its kerberos credentials to authenticate to the database instead of a login/password. When kerberos is enabled, the "Login ID" field is treated as the kerberos user principal name. The password, if supplied, is used to authenticate the user principal with the KDC. The password field may be left blank if the keytab set during appliance configuration contains keys for the user principal.

- Login ID — The user login this connector will use to connect to the database (not applicable
  to Kerberos Authentication).

- Password — The password associated with the Login ID or
  Username. (This password is stored encrypted.)

- Principal Name - (Kerberos Authentication only) The name of the Kerberos user principal to use when authenticating with the KDC. The realm portion of the principal may be omitted
  if it matches the configured default realm.

- Service Principal - (Sybase with Use Kerberos Authentication only) The name of the Sybase service instance.

- Port — The TCP port of the server.

- SID — (Oracle only) Oracle System ID (SID).

- Instance Name — (MS SQL Server only) The name of the instance. This is optional. If the instance name is specified, the connector ignores the specified "Port" and attempts to connect to the "SQL Server Browser Service" on port 1434 to retrieve the connection information for the SQL Server instance. If the instance name is provided, be sure to make exceptions in the firewall for port 1434 as well as the particular port that the SQL Server instance
  listens to.

- Custom Driver Name — (Generic only) The name of the
  JDBC driver class, including Java package name.

- JDBC URL — (Generic and Advanced connector mode for Oracle, MS SQL Server, and Sybase only) The custom
  JDBC URL, typically including hostname/IP and port number.

All database types have a Test Connection button at the bottom left of the New Connector window. We highly recommend that you test your connection before you save it. Do so before you leave this window. When you click Test Connection, Delphix uses the information in the form to attempt a database connection. When finished, a status message appears indicating success or failure.

File Connectors

The values that appear correlate to the File Type you select.

- Connector Name — The name of the file connector (specific to your Delphix application and unrelated to the file itself).
- Connection Mode — SFTP, FTP
- Path — The path to the directory where the file(s) are located.
- Server Name — The name of the server used to connect to the file.
- Port — The port used to connect to the server.
- User Name — The user name to connect to the server.
- Password — (non-Public Key Authentication only) The associated password for the server.
- Public Key Authentication — (Optional) (Only appears for SFTP.) Check this box to specify a public key. When you check this box, the Available Keys dropdown appears. Choose a key from the dropdown. See Delphix Masking APIs for information on uploading public keys to the masking engine.

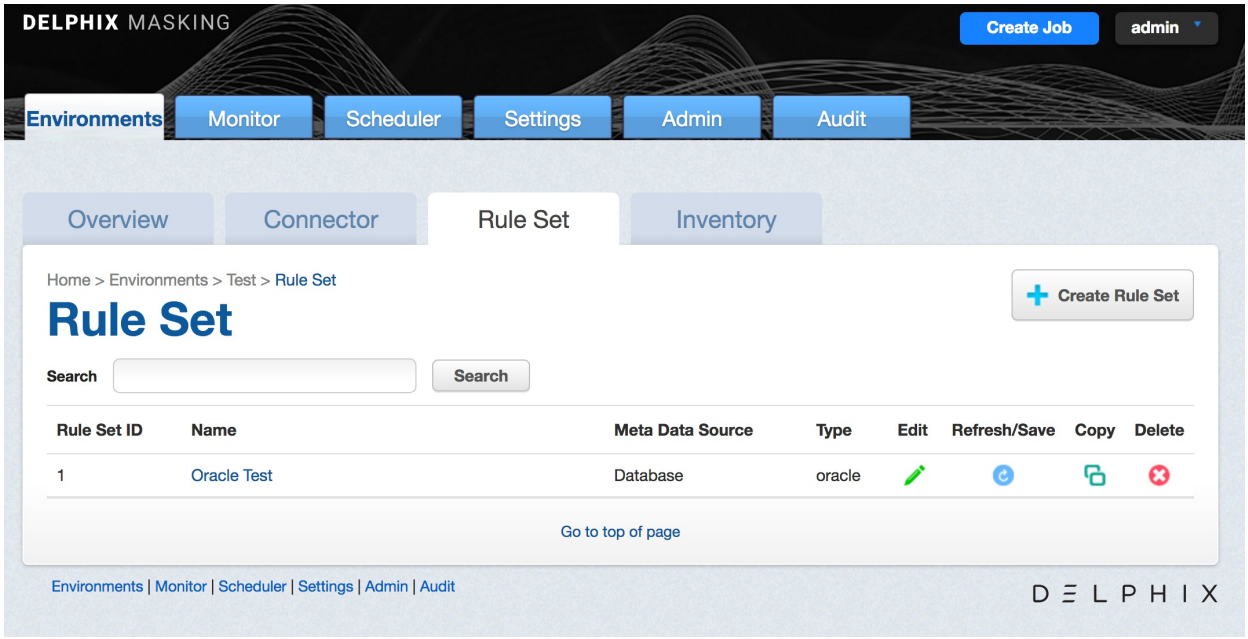
  Note:
  If you plan to do on-the-fly masking then you will need to create a separate environment and connector to be the source for the files to be masked. The masked files will get put into the directory pointed to by the connector you created previously (the target). However, the file path specified in the connector of the target rule set must point to an existing file the target directory. It does not have to be a copy of the file, just an entry in the directory with the same name. It will be replaced by the masked file.

Managing Rule Sets

This section describes how Rule Sets can be created, edited, and removed.

The Rule Sets Screen

From anywhere within an Environment, click the Rule Set tab to display the Rule Sets associated with that environment. The Rule Sets screen appears. If you have not yet created any rule sets, the Rule Set list is empty.



The Rule Sets screen contains the following information and actions:

- Rule Set ID — The numeric ID of the rule set used to refer to the rule set from the Masking API.
- Name — The name of the rule set.
- Meta Data Source — The type of rule set. One of Database, File, or Mainframe.
- Type — The specific type of rule set.
- Edit — Edit the rule set. See more details below.

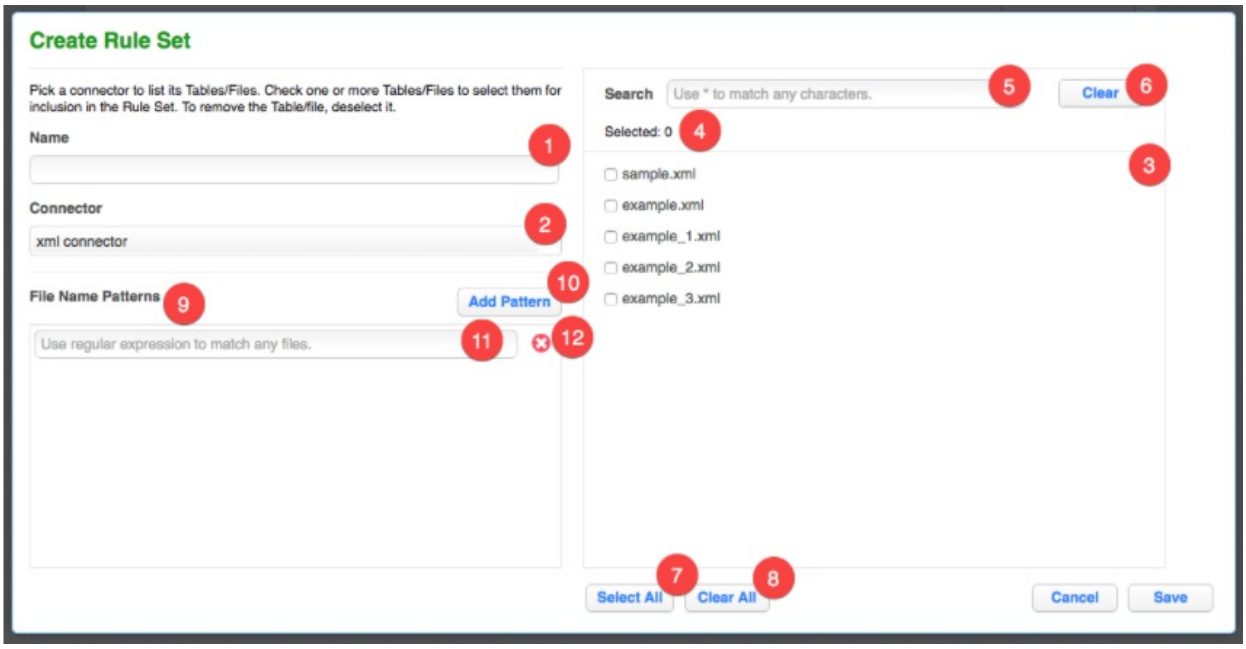
- Refresh/Save — Refresh the rule set. Only applies to Database rule sets. See more details below.
- Copy — Copy the rule set. See more details below.
- Delete — Delete the rule set. See more details below.

The rule sets on the screen can be sorted by the various informational fields by clicking on the respective field.

The Create/Edit Rule Set Window

In the upper right-hand corner, click the Create Rule Set button.

The Create Rule Set window appears.



1	Rule Set Name Input Field When editing an existing rule set, this field will be filled with the existing rule set name by default.
2	Connector List When creating a new rule set, all available connectors will be listed here. When editing an existing rule set, only the connector currently in use will appear.

3	Table or File List If a database connector is selected in the connector list, all available tables in the database schema associated with the connector will appear in this list. If a file connector is selected, all available files in the directory associated with the connector will appear in this list.
4	Selected Table or File Number Displays how many tables or files you have selected.
5	Search Query Input Field You can enter a search query here. After typing the search query, press ENTER to execute the search query. INFO: search query • Use * to match any characters in the names of tables or files. • If you have selected a table or file before searching and it is not in the search results, it will not be included in the rule set. You can add back the table or file by removing the search query.
6	Clear Search Button Click to remove any search query.
7	Select All Button Click to select all tables or files in the table or file list.
8	Clear All Button Click to deselect all tables or files in the table or file list.
9	File Name Patterns Editor

	This editor will appear only when the selected connector is a file connector.
10	Add File Pattern Button Click to add a new file pattern entry below.
11	File Pattern Input Field Enter the file pattern here.
12	Remove File Pattern Button Click to remove a file pattern.

Creating a Rule Set

To create a new rule set:

1. Click on the name of an Environment, and then click the Rule Set tab.
2. In the upper right-hand corner of the Rule Set screen, click Create Rule Set.
3. The Create Rule Set screen lets you specify which tables belong in the rule set.
4. Enter a name for the new Rule Set.
5. Select a Connector name from the drop-down menu.
6. The list of tables for that connector appears. If you have not yet created any connectors, the list is empty. Click individual table names to select them, or click Select All to select all the tables in the connector. See "Create/Edit Rule Set Window" for a description of the screen and other options.
7. Click Save.

You may then need to define the Rule Set by modifying the table settings

as described in "Modifying Tables in a Rule Set" below.

For example:

- For a table in a database rule set, you may want to filter data from the table.
- For a file in a file or mainframe rule set, you must select a File Format to use.

Refreshing a Rule Set

Refreshing a rule set will result in the columns in the tables in the rule set being rescanned. As a result, the inventory associated with the rule set will also be refreshed, but any pre-existing algorithm assignments will be retained.

To refresh a rule set:

1. Click the Refresh/Save icon to the right of the rule set on the Rule Set screen.
2. The Refresh/Savet icon will turn to an hour glass as the the associated tables are rescanned.
3. After the refresh is complete, the Refresh/Savet icon will return to the circular arrow.

Copying a Rule Set

If you copy a Rule Set, the inventory associated with that Rule Set will also be copied. Also, any filter conditions defined for that Rule Set will be copied.

To copy a rule set:

1. Click the Copy icon to the right of the rule set on the Rule Set screen.
2. The Copy Rule Set window appears.
3. Enter a Name for the new rule set.
4. Click Save.

5. Modify the rule set as you want, using the procedures described above.

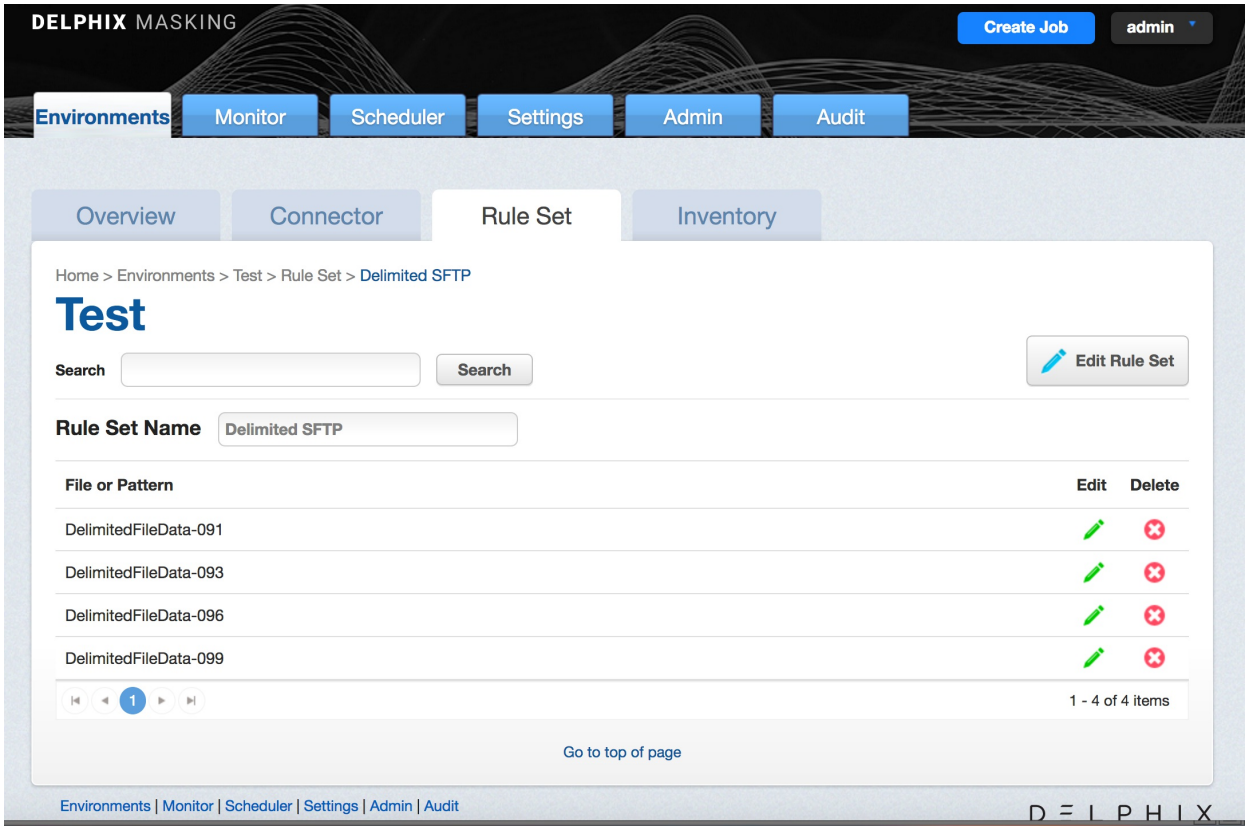
Deleting a Rule Set

If you delete a Rule Set, the inventory associated with that Rule Set will also be deleted. Also, any filter conditions defined for that Rule Set will be deleted.

To delete a rule set, click the Delete icon to the right of the rule set on the Rule Set screen.

The Rule Set Screen

From the Rule Set tab, click on a rule set to display the tables or files in the rule set. The Rule Set screen appears.



The Rule Set screen contains the following information and actions:

- Table or File or Pattern — The name of the table or file/file pattern in the rule set.

- Edit — Edit the table or file in the rule set. See more details below.
- Delete — Delete the table or file from the rule set.

For rule sets with a large number of tables or files, the Rule Set screen will be displayed on pages which can be navigated by the controls at the bottom of the list on the page. The tables or files displayed may also be filtered using the Search field and button.

Editing/Modifying a Rule Set

To edit a rule set:

1. Click the Edit icon to the right of the rule set on the Rule Set screen.
2. Click the Edit Rule Set button towards the top.
3. The Create Rule Set screen appears. This screen lets you specify which tables belong in the rule set.
4. Modify the rule set as you want, using the preceding procedures.

Removing a Table or File

To remove a table or file from a rule set:

1. From the Rule Set screen, click the name of the desired rule set.
2. Click the red delete icon to the right of the table or file you want to remove.

!!! note "INFO"
If you remove a table/file from a rule set and that table/file has an inventory, that inventory will also be removed.

Modifying Tables in a Rule Set

The features in this section are disabled for file and mainframe

rule sets.

You can modify tables in a rule set as follows:

Logical Key

If your table has no primary keys defined in the database, and you are using an In-Place strategy, you must specify an existing column or columns to be a logical key. This logical key does not change the target database; it only provides information to Delphix. For multiple columns, separate each column using a comma. Note: If no primary key is defined and a logical key is not defined an identify column will be created.

To enter a logical key:

1. From the Rule Set screen, click the name of the desired rule set.
2. Click the green edit icon to the right of the table whose filter you wish to edit.
3. On the left, select Logical Key.
4. Edit the text for this property.
5. To remove any existing code, click Delete.
6. Click Save.

Edit Filter

Use this function to specify a filter to run on the data before loading it to the target database.

To add a filter to a database rule set table or edit a filter:

1. From the Rule Set screen, click the name of the desired rule set.
2. Click the green edit icon to the right of the table you want.
3. On the left, select Edit Filter.

4. Edit the properties of this filter by entering or changing values in the Where field.

Be sure to specify column name with table name prefix (for example, customer.cust_id <1000).

To remove an existing filter, click Delete.

- Click Save.

Custom SQL

Use this function to use SQL statements to filter data for a table.

To add or edit SQL code:

1. From the Rule Set screen, click the name of the desired rule set.
2. Click the green edit icon to the right of the table you want.
3. On the left, select Custom SQL.
4. Enter custom SQL code for this table.

Delphix will run the query to subset the table based on the SQL you specify.

1. To remove any existing code, click Delete.
2. Click Save.

Table Suffix

If you have tables with names that change monthly, for example tables that are appended with the current date, you can set a table suffix for a rule set.

To set a table suffix for a rule set:

1. In the Rule Set screen, click the name of the desired rule set.

2. Click the green edit icon to the right of the table for which you wish to set the suffix.

3. On the left, select Table Suffix.

4. The Original Table Name will already be filled in.

5. (Optional) Enter a Suffix date Pattern (for example, mmyy).

6. (Optional) Enter a Suffix Value, if you want to append a specific value.

7. (Optional) Enter a Separator (for example, _). This value will be inserted before the suffix value (for example, tablename_0131).

8. Click Save.

Add Column

Use this function to select a column or columns from a table when you don't want to load data to all the columns in a table.

To add a column to a database rule set table or edit a column:

1. From the Rule Set screen, click the name of the desired rule set.

2. Click the green edit icon to the right of the table you want.

3. On the left, select Add Column.

4. Select one or more column names to include in the table. To remove a column, deselect it.

5. You can also choose Select All or Select None.

6. Select Save.

Join Table

Use this function to specify a SQL join condition so that you can define primary key/foreign key relationships between tables.

To define or edit the join condition for a table:

1. From the Rule Set screen, click the name of the desired rule set.
2. Click the green edit icon to the right of the table you want.
3. On the left, select Join Table.
4. Edit the properties for this join condition.
5. To remove an existing join condition, click Delete.
6. Click Save.

List

Use this function to select a list to use for filtering data in a table.

To add or edit a list:

1. From the Rule Set screen, click the name of the desired rule set.
2. Click the green edit icon to the right of the table you want.
3. On the left, select List.
4. Edit the text file properties for this list.
    i. Select a column.
    ii. Enter or browse for a filename.
    iii. Files that have already been specified appear next to Existing File.
5. To remove an existing list file, click Delete.
6. Click Save.

Creating a Ruleset For File Formats

Once you create a ruleset with a file or set of files, you will need to assign those files to their appropriate file format.

This is accomplished by editing the ruleset. Click on the edit button for the file the Edit File window will appear with the file name. From the format drop-down select the proper format for the file.

- If the file is a Mainframe data sets file with a copybook you will see a checkbox to signify if the file is variable length.
- For all other file types, select the end-of-record to let Delphix know whether the file is in windows/dos format (CR+LF) or Linux format (LF).
- If the file is a delimited file you will have a space to put in the delimiter.
- If there are multiple files in the ruleset you will have to edit each one individually and assign it to the appropriate file format.

Managing Inventories

An inventory describes all of the data present in a particular data source and defines the methods which will be used to secure it. Inventories typically include the table or file name, column/field name, the data classification, and the chosen algorithm.

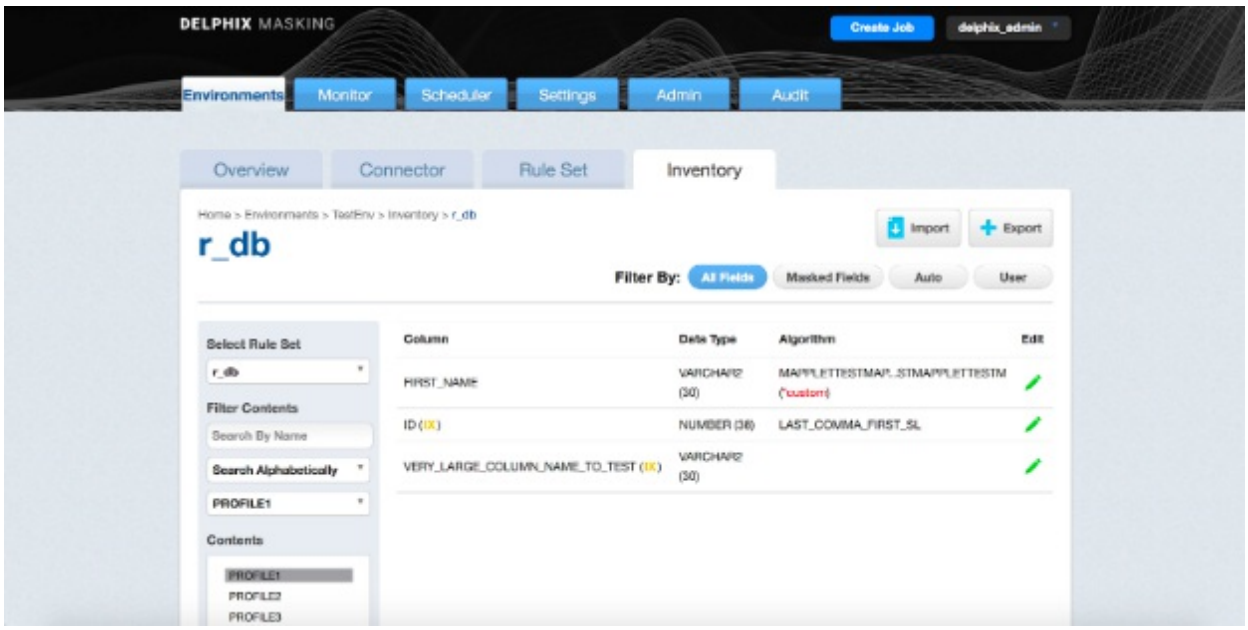
The Inventory Screen

From anywhere within an environment, click the Inventory tab to see the Inventory Screen. This displays the inventory for the environment's rule sets.

Inventory Settings

To specify your inventory settings:

1. On the left-hand side of the screen, select a Rule Set from the drop-down menu.
2. Below this, Contents lists all the tables or files defined for the rule set.



3. Select a table or file for which you want to create or edit the inventory of sensitive data. The Columns or Fields for that specific table or file appear.

4. If a column is a primary key (PK), Foreign Key (FK), or index (IDX), an icon indicating this will appear to the Right of the column name. If there is a note for the column, a Note icon will appear. To read the note, click the icon.

5. If an algorithm associated with a column is a custom algorithm (formerly known as Mapplet) then (custom) in red text will appear after the algorithm name.

6. If you selected a table, metadata for the column appears: Data Type and Length (in parentheses). This information is read-only.

7. Choose how you would like to view the inventory:

    - All Fields — Displays all columns in the table or all fields in the file (allowing you to mark new columns or fields to be masked).
    - Masked Fields — Filters the list to just those columns or fields that are already marked for masking.

8. Choose how to determine whether to mask/unmask a column:

- Auto — The default value. The profiling job can determine or update the algorithm assigned to a column and whether to mask the column.
- User — The user's choice overrides the profiling job. The user manually updates the algorithm assignment, mask/unmask option of the column. The Profiler will ignore the column, so it will not be updated as part of the Profiling job. In order to use the Secure Lookup algorithm, the user would select it as a user-defined algorithm and assign it to the specific column. Secure Lookup automates the creation of a secure lookup algorithm by building a list of replacement values based on the existing unique values in the target column and creating a secure lookup using those values. In that respect, it is simply shuffling the values.

Managing a Database inventory

The following sections apply to databases.

Setting Column Criteria for a Table

Note:
You must select a database from the Select Rule Set drop-down menu on the left, not a file system.

To set criteria for sensitive columns:

1. Click the green edit icon to the right of a column's name.
2. To mask the selected column, select the Mask check box.

- If you do not want to mask this column, clear this check box.

1. From the Domain drop-down menu, select the appropriate sensitive data element type for the column.
2. The Delphix masking engine defaults to a Masking Algorithm as specified in the Settings screen. If necessary, you can override the default algorithm for a column.

- To select a different masking algorithm, choose one from the Algorithm dropdown.
- To identify a custom algorithm, a text as custom in parenthesis will appear with the algorithm name.

For detailed descriptions of these algorithms, please see [Securing Sensitive Data - Configuring Your Own Algorithms].

If you select a DATESHIFT algorithm and you are not masking a datetime or timestamp column, you must specify a Date Format. (This field only appears if you select a DATESHIFT algorithm from the Masking Algorithm dropdown.) For a list of acceptable formats, click the Help link for Date Format. The default format is yyyy-MM-dd.

1. Select the Row Type according to its purpose, using "All Row" as a convention for all rows.
   If you need to create a row type (for example, if filter conditions are required), see Row Types and Creating New Row Types for a Table next.
2. Select an ID Method:

- Auto — The default value. The profiling job can determine or update whether to mask a column.
- User — The user decides whether to mask/unmask a column. The user's choice overrides the profiling job. (The user masking is done after the profiling job is finished.)

1. You can add/remove notes in the Notes text field.
2. When you are finished, click Save.
   You must click Save for any edits to take effect.

Creating a New Row Type

1. From an Environment's Inventory tab, click +Row Types in the upper right.
   The Row Type window appears, listing existing row types.
2. Click + Add a Row Type. The Add Row Type window appears.
3. Name the Row Type according to its purpose. For example, if you want to subset the rows to only take rows with addresses, you can name this row type "Address Rows".
4. To limit the masking to a subset of rows, specify an appropriate Where Clause.
5. Click Save.

Managing a File Inventory

Setting Field Criteria for a file

To set criteria for sensitive fields:

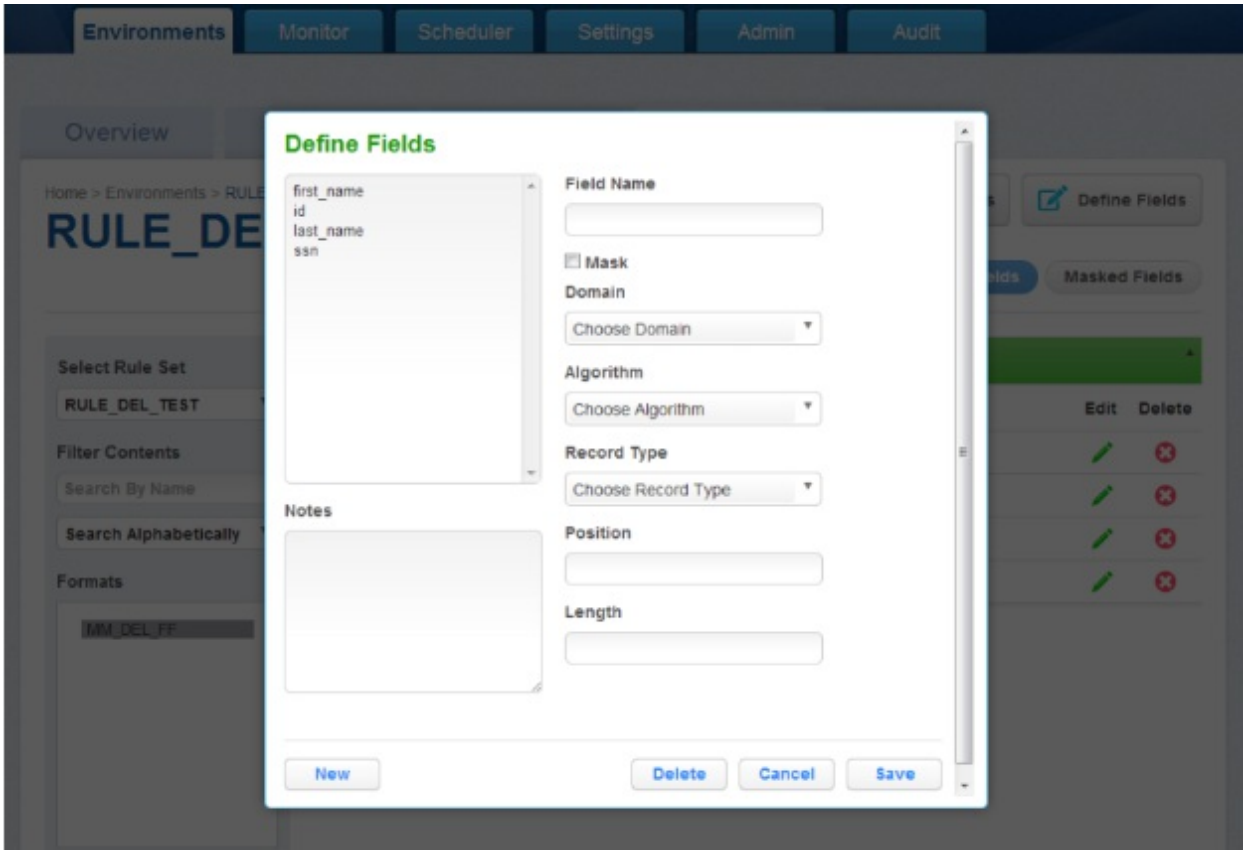
1. From an Environment's Inventory tab, click the green edit icon to the right of the field you want.
2. To mask this field, check the Mask check box (in the View Inventory pane).
3. Clear this check box if you do not want to mask this field.
4. Choose the appropriate sensitive data element type for the field from the Domain drop-down.
5. Delphix defaults to a masking Algorithm as specified in the Settings screen. If necessary, you may override the default algorithm for a field.
6. To select a different masking algorithm, choose one from the Algorithm drop-down.
7. Choose a Record Type from the drop-down.
8. Click Save when you are finished.

Note:
You must click Save for any edits to take effect.

Defining fields

To create new fields:

1. From an Environment's Inventory tab, click Define fields to the far right.
   The Edit Fields window appears.



2. Edit the fields as described in Setting Field Criteria for a File.
3. When you are finished, click New to create a new field, or click Save to update an existing field.

Adding Record Types for files

To add a new Record Format:

1. In the upper right-hand corner of an environment's Inventory tab, click Record Types. The Record Type window appears.
2. Click +Add a Record Type towards the bottom of the window. The Add Record Type window appears.
3. Enter values for the following fields:

- Record Type Name — A free-form name for this record format.
- Header/Body/Trailer — If the file has header or trailer records, you will need to create file formats for them. Select the appropriate type. Delphix allows for masking of multiple headers, multiple trailers, and multiple types of body records.
- Record Type ID — (optional) For body records, specify the value of the record type code or other identifier that allows Delphix to identify records that qualify as this record type.
- Position # — (optional) Specify the field number (for delimited files) or the character position number (for fixed files) of the beginning of the Record Type Identifier within the

data record.
- Length # — (optional) For fixed files, specify the length of the Record Type Identifier within the data record.

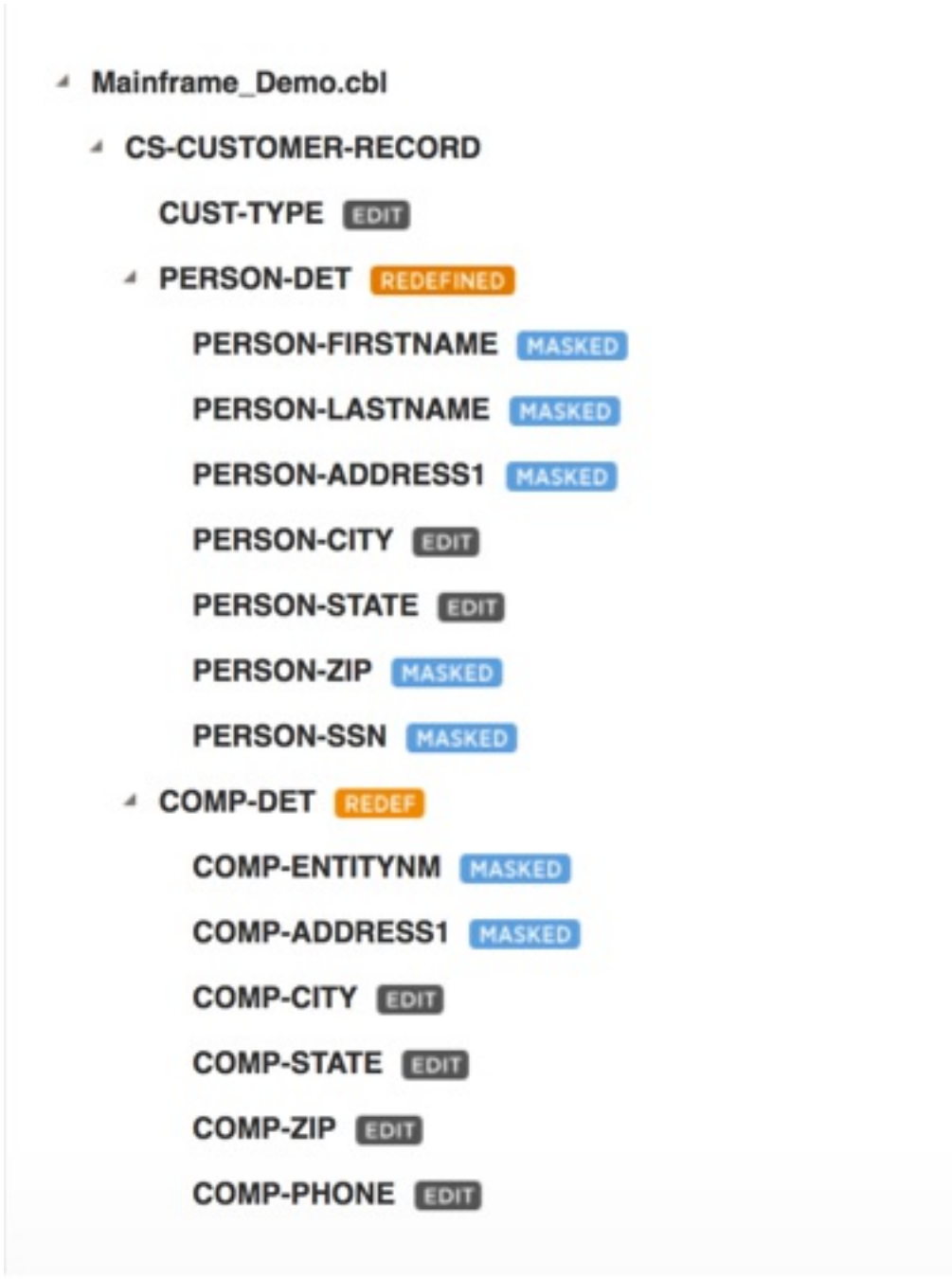
1. Click Save when you are finished.

Redefine Conditions

For Mainframe files, the inventory also allows for the entry of Redefine Conditions, which are used to handle any occurrences of COBOL's REDEFINES construct that might appear in the Copybook. In COBOL, the REDEFINES keyword allows an area of a record to be interpreted in multiple different ways. In the example below, for instance, each record can hold either the details of a person (PERSON-DET) or the details of a company (COMP-DET).

```
01  CS-CUSTOMER-RECORD.
    05  CUST-TYPE                      PIC X(1).
    05  PERSON-DET.
        10  PERSON-FIRSTNAME           PIC X(20).
        10  PERSON-LASTNAME            PIC X(40).
        10  PERSON-ADDRESS1            PIC X(50).
        10  PERSON-CITY                PIC X(20).
        10  PERSON-STATE               PIC X(5).
        10  PERSON-ZIP                 PIC X(10).
        10  PERSON-SSN                 PIC S9(9) COMP-3.
    05  COMP-DET                       REDEFINES PERSON-DET.
        10  COMP-ENTITYNM              PIC X(53).
        10  COMP-ADDRESS1              PIC X(50).
        10  COMP-CITY                  PIC X(20).
        10  COMP-STATE                 PIC X(5).
        10  COMP-ZIP                   PIC X(10).
        10  COMP-PHONE                 PIC X(12).
```

Depending on which group is present, different masking algorithms may need to be applied. Below is the inventory corresponding to this copybook, which allows algorithms to be selected separately for each group.



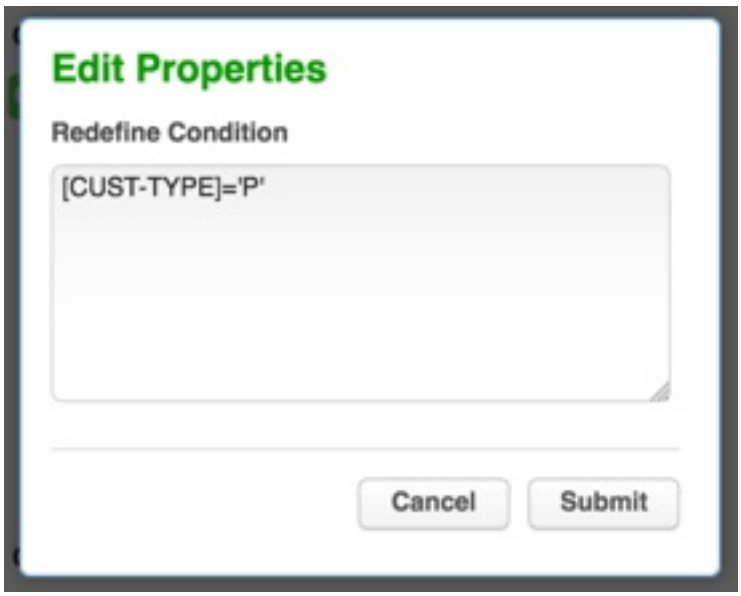
In order to do any masking however, the masking engine must be able to determine, for each record, which fields should be read, so that the correct algorithms can be applied. In order to do this, the masking engine uses Redefine Conditions, which are specified in the inventory. Redefine Conditions are boolean expressions which can reference any fields in the record when they are evaluated.

In the example copybook above, the field CUST-TYPE is used to indicate which group is present. If CUST-TYPE holds a 'P', a PERSON-DET group is present, and if it holds a 'C', COMP-DET is present. This can be expressed in the inventory by specifying a Redefine Condition with the value [CUST-TYPE]='P' . This expression indicates that, for each record read from the source file during the masking job, the value of the field CUST-TYPE should be

read and compared against the string 'P'. If it is equal, the masking engine will read from the record the fields subordinate to PERSON-DET, and will apply any masking algorithms specified on those fields. Similarly, a Redefine Condition with the value [CUST-TYPE]='C' should be applied to the COMP-DET field.
Exactly one of the conditions should evaluate to 'true' for each group of redefined fields. For example, a copybook might have fields A, B REDEFINES A, and C REDEFINES A. Of the Redefine Conditions attached to A, B, and C, one and only one should evaluate to true for each record.

Entering a Redefine Condition

1. Click on the orange REDEFINED or REDEF button next to the redefined or redefining field
2. Enter a condition in the dialog box which appears. This is the expression, which, when it evaluates to true, causes the subordinate fields to be read and, if they have algorithms assigned, masked.



3. Click Submit.

Format of Redefine Conditions

Redefine Conditions allow fields to be compared against either number or string literals. Square brackets enclosing a field name indicate a variable, which takes on the value of the named field:

```
[Field1] = 'An example String'
```

String literals can be enclosed in either single or double quotes. For fields that are numeric (e.g. PIC S99V9), the operators <, <=",">, and >= can be used in addition to the =operator, e.g.

```
[Field2] <= -10.5
```

Also, conditions can be joined using AND, OR, and NOT to form more complex conditions:

```
([Field3] > 2.5 AND [Field3] < 10) OR NOT [FIELD4] = 'Z'
```

Importing and Exporting an Inventory

To export an inventory:

1. Click the Export icon at the upper right. The Export Inventory popup appears with the name of the currently selected Rule Set as the Inventory Name and a corresponding .csv File Name.
2. Click Save.

A status popup appears. When the export operation is complete, you can click on the Download file name to access the inventory file

To import an inventory:

1. In the upper right-hand corner, click the Import icon. The Import Inventory popup appears.
2. Click Select to browse for the name of a comma-separated (.csv) file.
3. Click Save.

The inventory you imported appears in the Rule Set list for this environment.

!!! info

The format of an imported.csv file must exactly match the format of the exported inventory. If you plan to import an inventory, before importing the inventory, you should export it and then update the exported file as needed before you import it.

Managing File Formats

##File formats

Unlike databases files for the most part do not have built in metadata to describe the format of the fields in the file. You must provide this to Delphix so it can update the file appropriately. This is done through the settings tab where you will see a menu item on the left for File Format. Select File Format and you will see options to create a file format or input a file format. This will depend on the type of file and how you want to let Delphix know the format of the file.

##Mainframe and XML files

For Mainframe data sets, you can specify the file format via Input Format option which will import the copybook directly into Delphix. You can input this file from SFTP or FTP. Please select Copybook as the Import Format Type.
For XML files you can also input the file format with the input format option. You can use the file you want to mask as the format. Delphix will input the format of the file directly. You can input this file from SFTP or FTP. Please select XML as the Import Format Type.

##Delimited, Excel, Fixed files

For Delimited, Excel, and Fixed files you can either manually create the format of the file yourself, or you can input a text file which describes the structure of the file to Delphix. To input the file format for delimited or Excel files create a text document with the column names each on its own line. For example:

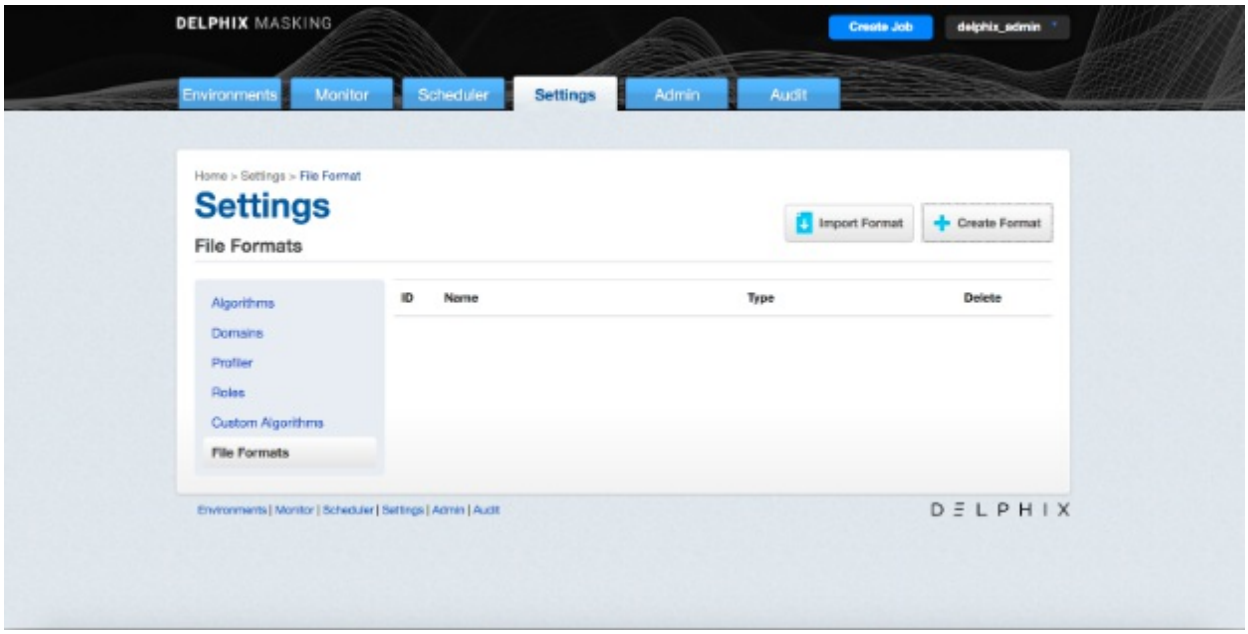
- Name
- Address
- City
- State

To input the file format for fixed files create a text document with the column names and the length of each column on its own line. For example:

- Name,25
- Address,40
- City,20
- State,2

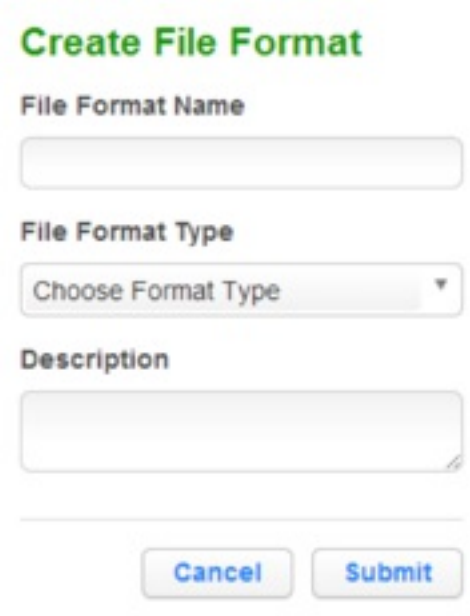
Then input this file as the file format. The name of the text file will be the name of the file format. To create a format manually, you can just click the create format button and give the

format a name. We will input the details of the format a little later in this document.



1. Click Create Format in the upper right. The Create File Format window appears.
2. Enter a File Format Name.
3. Choose a File Format Type:
    - Delimited File
    - Excel Sheet
    - Fixed Width File

    Note:
    Creating a Copybook or XML file format is not supported. These formats must be imported instead.
4. Optionally, enter a Description.
5. Click Submit.



To Import a New File Format

1. Click Import Format at the upper right. The Import File Format window appears.
2. Select an Import File Type.

For a Format Type of Copybook or XML

1. Select a Connection Mode.
2. Fill out the required fields of the selected Connection Mode.
3. Click Browse.
4. Click the Select button to the right of the desired import file format.
5. Enter a Logical Name.
6. Click Submit.

For a Format Type of Delimited File, Excel sheet, or Fixed Width File

1. Click Select.

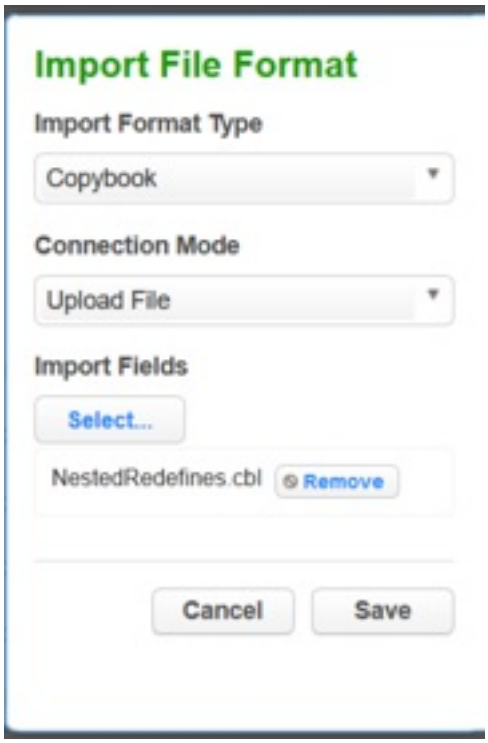
2. Browse for the file from which to import fields.

3. Click Save.

   Note:

- The file must have NO header.
- Make sure there are no spaces or returns at the end of the last line in the file.
- To be masked, the field names must be in the same order as they are in the file.

Removing a Selected File



If you accidentally selected an incorrect file, simply click Remove button to the right of the file and repeat selection steps above.

Samples

The following is sample file content for Delimited or Excel file formats. With these formats just the field name is provided. Notice there is no header and only a list of values.

```
First_Name
Last_Name
DOB
SSN
Address
City
State
Zip_Code
```

The following is sample file content for Fixed Width format. In this format the field name is followed by the length of the field, separated by a comma. Notice there is no header and only a list of values.

```
First_Name,20
Last_Name,30
DOB,10
SSN,11
Address,30
City,20
State,2
Zip_Code,10
```

To Delete a File Format

1. Click the Delete icon to the right of the File Format name.
2. File inventory is based on file format. Therefore, if you make a change to a file inventory, that change applies to *all* files that use that format.
3. You can only add or delete a file format; you cannot edit one.

Assigning a File Format to a files

Once you create a ruleset with a file or set of files, you will need to assign those files to their appropriate file format. This is accomplished by editing the ruleset. When you click on the edit button for the file a popup screen called edit file will appear with the file name. There will be a dropdown for the format so you can select the proper format for the file. If the file is a Mainframe data sets file with a copybook you will see a checkbox to signify if the file is

variable length. For all other file types, select the end-of-record to let Delphix know whether the file is in windows/dos format (CR+LF) or Linux format (LF). If the file is a delimited file you will have a space to put in the delimiter. If there are multiple files in the ruleset you will have to edit each one individually and assign it to the appropriate file format.

Discovering Your Sensitive Data

After connecting data to the masking service, the next step is to discover which of the data should be secured. This sensitive data discovery is done using two different methods, column level profiling and data level profiling.

Column Level Profiling
Column level profiling uses regular expressions (regex) to scan the metadata (column names) of the selected data sources. There are several dozen pre-configured profile Expressions (like the one below) designed to identify common sensitive data types (SSN, Name, Addresses, etc). You also have the ability to write your own profile Expressions.

First Name Expression <([a-z][a-z0-9])\b[^>]>(.*?)<\/1>

Data Level Profiling
Data level profiling also uses regex, but to scan the actual data instead of the metadata. Similar to column level profiling, there are several dozen pre-configured Expressions (like the one below) and you can add your own.

Social Security Number Expression <([a-z][a-z0-9])\b[^>]>(.?)<\/1>*

For both column and data level profiling, when a data item is identified as sensitive, Delphix recommends/assigns particular masking algorithms to be used when securing the data. The platform comes with several dozen pre-configured algorithms which are recommended when the profiler finds certain sensitive data.

Out of the Box Profiling Settings

The Delphix Platform comes out of the box with over 50 profile Expressions to help you discover over 30 types (account numbers, addresses, etc.) of sensitive data.

Account Numbers

An account number is the primary identifier for ownership of an account, whether a vendor account, a checking or brokerage account, or a loan account. An account number is used whether or not the identifier uses letters or numbers. Below are the profile Expressions Delphix uses to identify account numbers:

Expression Name	Domain	Expression Level	Expression
Account Number	ACCOUNT_NO	Column	`(?>(acc(oun|n)?t)_?(num(ber)?|nbrjno)?)(?!\w\*(ID|type))`

Physical Addresses

Below are the profile Expressions Delphix uses to identify physical addresses:

Expression Name	Domain	Expression Level	Expression
Address	ADDRESS	Column	`^(?:(?!postalcode|city|state|country|em(l|ln|lin|line)?_?2{1}|ID).)*addre?_?(?:(?!city|state|country|email|(l|ln|lin|line)?_?2{1}|ID).)*$`
Street Address	ADDRESS	Column	`(?>(str(eet)?_?addre?s?s?|street))(?!\w*(ID|type))`
Data -			`(.*[\s]+b(ou)?|(e)?v(ar)?d[\d]*.*)|[\s]+st[.]?(reet)?[\s]*.*)|(.*`

Address	ADDRESS	Data	`[\s]+ave[.]?(nue)?[\s]*.*)|(.*[\s]+r(oa)?d[\s]*.*)|(.*[\s]+|(a)?n[\s]*.*)|(.[\s]+cir(cle)?[\s]*.*1`
Address Line2 - before	ADDRESS_LINE2	Column	`^(?:(?!email|ID).)*(l|ln|lin|line)2{1}_?addre?s?s?(?:(?!email|ID).)*$`
Address Line2 - after	ADDRESS_LINE2	Column	`^(?:(?!email|ID).)*addre?s?s?_?(l|ln|lin|line)?_?2{1}(?:(?!email|ID).)*$`
Data - Address Line 2	ADDRESS_LINE2	Data	`(.*[\s]*ap(ar)?t(ment)?[\s]+.*)|(.*[\s]*s(ui)?te[\s]+.*)|(c(are)?[\s]*[\\\\]?[/]?o(f)?[\s]+.*)`

Beneficiary ID

Below are the profile Expressions Delphix uses to identify beneficiary IDs:

Expression Name	Domain	Expression Level	Expression
Beneficiary Number	BENEFICIARY_NO	Column	`(?>(bene(ficiary)?)_?(num(ber)?|nbr|no))(?!\w*ID)1`
Beneficiary ID	BENEFICIARY_NO	Column	`(?>(bene(ficiary)?)_?id)`

Biometrics

Below are the profile Expressions Delphix uses to biometric data:

Expression Name	Domain	Expression Level	Expression
Biometric	BIOMETRIC	Column	biometric

Certificate ID

Below are the profile Expressions Delphix uses to identify certificate IDs:

Expression Name	Domain	Expression Level	Expression
Certificate Number	CERTIFICATE_NO	Column	`(?>cert(ificate)?_?(num(ber)?|nbr|no|id))`
Certificate ID	CERTIFICATE_NO	Column	`(?>cert(ificate)?_?id)`

City

Below are the profile Expressions Delphix uses to identify cities:

Expression Name	Domain	Expression Level	Expression
City	CITY	Column	`ci?ty(?!\w*ID)`

Country

Below are the profile Expressions Delphix uses to identify countries:

Expression Name	Domain	Expression Level	Expression
Country	COUNTRY	Column	`c(ou)?nty(?!\w*ID)`

Credit Card

Below are the profile Expressions Delphix uses to identify credit cards:

Expression Name	Domain	Expression Level	Expression
Card Number	CREDIT CARD	Column	`(?>ca?rd_?(num(ber)?|nbr|no)?)(?!\w*ID)`

Credit Card Number	CREDIT CARD	Column	`(?>cre?di?t_?(ca?rd)?_?(num(ber)?|nbr|no)?)(?!\w*ID)`
Data - Credit Card	CREDIT CARD	Data	`^(?:3[47][0-9]{13}|4[0-9]{12}(?:[0-9]{3})?(?:[0-9]{3})?|(?:5[1-5][0-9]{2}|222[1-9]|22[3-9][0-9]|2[3-6][0-9]{2}|27[01][0-9]|2720)[0-9]{12}|6(?:(011|5[0-9][0-9])[0-9]{2}|4[4-9][0-9]{3}|2212[6-9]|221[3-9][0-9]|22[2-8][0-9]{2}|229[0-1][0-9]|2292[0-5])[0-9]{10}?(?:[0-9]{3})?|3(?:0[0-5,9]|6[0-9])[0-9]{11}|3[89][0-9]{14}?(?:[0-9]{1,3})?)$`

Customer Number

Below are the profile Expressions Delphix uses to identify customer IDs:

Expression Name	Domain	Expression Level	Expression
Customer Number	CUSTOMER_NUM	Column	`(?>(cu?st(omer|mr)?)_?(num(ber)?|nbr|no)?)(?!\w*ID)`

Date of Birth

Below are the profile Expressions Delphix uses to identify dates of birth:

Expression Name	Domain	Expression Level	Expression
Birth Date	DOB	Column	`(?>(bi?rth)_?(date?|day|dt))(?!\w*ID)`
Birth Date1	DOB	Column	`(?>dob|dtofb|(day|date?|dt)_?(of)?_?(bi?rth))(?!\w*ID)`
Birth Date2	DOB	Column	`(?>b_?(date?|day))(?!\w*ID)`

Admission Date	DOB	Column	`(?>(adm(it|ission)?)_?(date?|day|dt))(?!\w*ID)`
Treatment Date	DOB	Column	`(?>(tr(ea)?t(ment)?)_?(date?|day|dt))(?!\w*ID)`
Discharge Date	DOB	Column	`(?>(ds|disc(h|harge)?)_?(date?|day|dt))(?!\w*ID)`

Driver License Number

Below are the profile Expressions Delphix uses to identify driver license numbers:

Expression Name	Domain	Expression Level	Expression
Drivers License Number	DRIVING_LC	Column	`(?>(dri?v(e?rs?e?)?)_?(license|li?c)?_?(num(ber)?|nbr|no)?)(?!\w*ID)`
Drivers License Number1	DRIVING_LC	Column	`(^license$|(license|li?c)_?(num(ber)?|nbr|no))(?!\w*ID)`

Email

Below are the profile Expressions Delphix uses to identify emails:

Expression Name	Domain	Expression Level	Expression
Email	EMAIL	Column	`^(?:(?!invalid).)*email(?!\w*ID)`
Data - Email	EMAIL	Column	`\b[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,6}\b`

First Name

Below are the profile Expressions Delphix uses to identify first names:

Expression Name	Domain	Expression Level	Expression
First Name	FIRST_NAME	Column	`(?>(fi?rst)_?(na?me?)|f_?name)(?!\w*ID)`
Middle Name	FIRST_NAME	Column	`(?>(mid(dle)?)_?(na?me?)|m_?name)(?!\w*ID)`

IP Address

Below are the profile Expressions Delphix uses to IP addresses:

Expression Name	Domain	Expression Level	Expression
IP Address	IP ADDRESS	Column	`(?>(ip_?addre?s?s?))(?!\w*(ID|type))`
Data - IP Address	IP ADDRESS	Data	`\b(?:(?:25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9]?[0-9])\.){3}(?:25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9]?[0-9])\b`

Last Name

Below are the profile Expressions Delphix uses to identify last names:

Expression Name	Domain	Expression Level	Expression
Last Name	LAST_NAME	Column	`^(?:(?!portal|ID).)*((la?st)_?(na?me?)|l_?name)(?:(?!portalname|ID).)*$`

Plate Number

Below are the profile Expressions Delphix uses to identify plate

numbers:

Expression Name	Domain	Expression Level	Expression
License Plate	PLATE_NO	Column	`^(?:(?!template|ID|type).)*(license|li?c)?_?plate_?(num(ber)?|nbr|no)?(?:(?!template|ID|type).)*$`

PO Box Numbers

Below are the profile Expressions Delphix uses to identify PO box numbers:

Expression Name	Domain	Expression Level	Expression
PO Box	PO_BOX	Column	`po_?box`
Data - PO Box	PO_BOX	Data	`po box|p\.o\`

Precinct

Below are the profile Expressions Delphix uses to identify precincts:

Expression Name	Domain	Expression Level	Expression
Precinct	PRECINCT	Column	`(>?precinct|prcnct)(?!\w*ID)`

Record Number

Below are the profile Expressions Delphix uses to identify record numbers:

Expression Name	Domain	Expression Level	Expression

Record Number	RECORD_NO	Column	`(?>rec(ord)?_?(num(ber)?|nbr|no))(?!\w*(ID|type))`

School Name

Below are the profile Expressions Delphix uses to identify school names:

Expression Name	Domain	Expression Level	Expression
School Name	SCHOOL_NM	Column	`(?>school_?na?me?)(?!\w*ID)`

Security Code

Below are the profile Expressions Delphix uses to identify security codes:

Expression Name	Domain	Expression Level	Expression
Security Code	SECURITY_CODE	Column	`(?>se?cu?r(i?ty?)?_?co?de?)(?!\w*ID)`

Serial Number

Below are the profile Expressions Delphix uses to identify serial numbers:

Expression Name	Domain	Expression Level	Expression
Serial Number	SERIAL_NM	Column	`(?>(ser(ial)?)_?(num(ber)?|nbr|no))(?!\w*ID)`

Signature

Below are the profile Expressions Delphix uses to identify signatures:

Expression Name	Domain	Expression Level	Expression
Signature	SIGNATURE	Column	`signature(?!\w*(ID|type))`

Social Security Number

Below are the profile Expressions Delphix uses to social security numbers:

Expression Name	Domain	Expression Level	Expression
Social Security Number	SSN	Column	`ssn(?!\w*ID)`
Data - SSN	SSN	Data	`\b(?!000)(?!666)[0-8]\d{2}[- ](?!00)\d{2}[- ](?!0000)\d{4}\b`

Tax ID

Below are the profile Expressions Delphix uses to identify tax IDs:

Expression Name	Domain	Expression Level	Expression
Tax ID Number	TAX_ID	Column	`tin$|^tin|_tin|tin_`
Tax ID Code or Number	TAX_ID	Column	`(ta?x)_?(id(ent)?)?_?((co?de?)|(num(ber)?|nbr|no))?`

Telephone Number

Below are the profile Expressions Delphix uses to identify telephone numbers:

Expression Name	Domain	Expression Level	Expression
Telphone or Contact Number	TELEPHONE_NO	Column	`(?>((tele?)?phone)|(co?nta?ct|tel)_?(num(ber)?|nbr|no))(?!\w*(ID|type))`
Data - Phone Number	TELEPHONE_NO	Data	`\(?\b[0-9]{3}\)?[-. ]?[0-9]{3}[-. ]?[0-9]{4}\b`
Fax Number	TELEPHONE_NO	Data	`(?>fax_?(num(ber)?|nbr|no)?)(?!\w*(ID|type))`

Vin Number

Below are the profile Expressions Delphix uses to identify vin numbers:

Expression Name	Domain	Expression Level	Expression
Vehicle	VIN_NO	Column	vehicle
VIN	VIN_NO	Column	`vin$|^vin|_vin|vin_`

Web Address

Below are the profile Expressions Delphix uses to identify web addresses:

Expression Name	Domain	Expression Level	Expression
Web or URL Address	WEB	Column	`(?>(url|web_?addre?s?s?))(?!\w*(ID|type))`
Data - Web Address	WEB	Data	`\b(?:(?:https?|ftp|file)://|www\.|ftp\.)[-A-Z0-9+&-@#/%=~_|$?!:,.]*[A-Z0-9+&-@#/%=~_\|$]`

ZIP Code

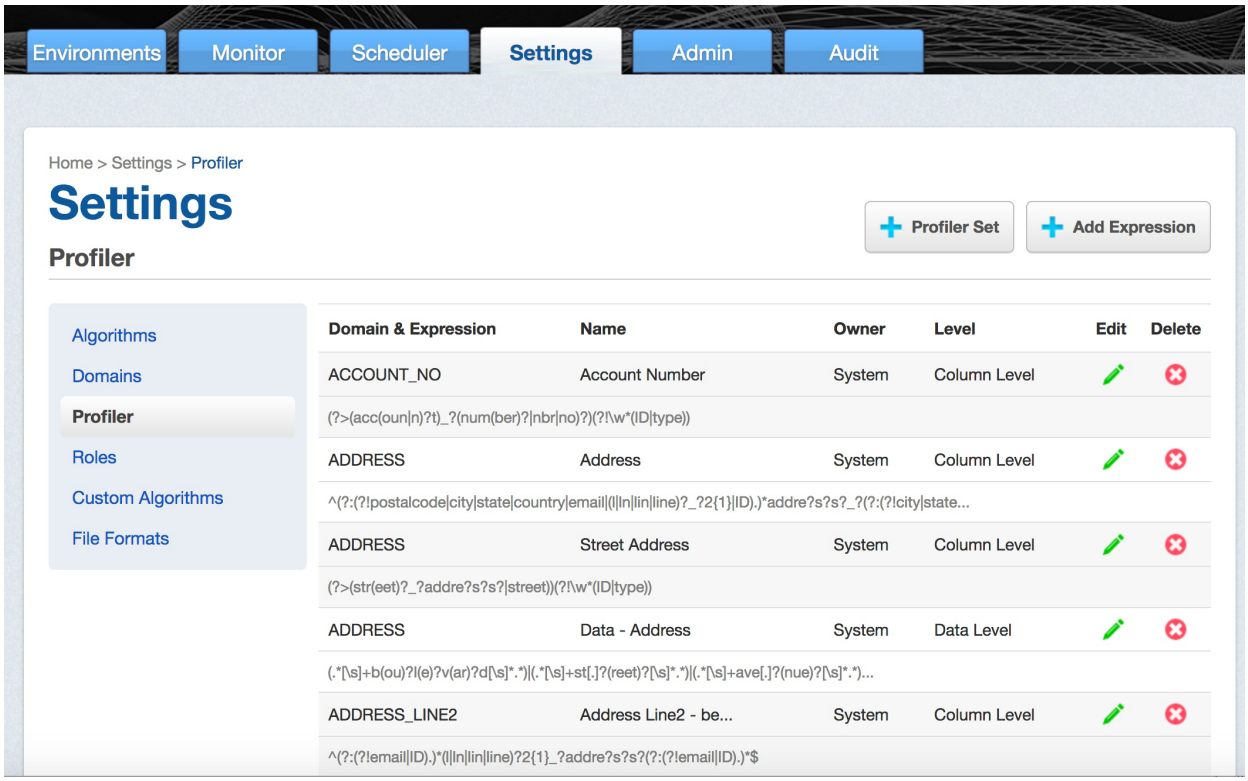
Below are the profile Expressions Delphix uses to identify zip codes:

Expression Name	Domain	Expression Level	Expression
zip or Postal Code	ZIP	Column	`(?>(zip|post(al)?)_?((co?de?)?4?))(?!\w*ID)`
Data - Zip Code	ZIP	Data	`1\b([0-9]{5})-([0-9]{4})\b`

Configuring Profiling Settings

In addition to using your Rule Set to determine the inventory of what to profile, a Profiling job uses Expressions to identify your sensitive data. You can add regular expressions to be used by Profiler Sets to the Profiler Settings.

To display the Profiler Settings, click on the Settings tab and select Profiler on the left-hand side of the page.



The Profiler Settings screen displays Expressions along with their Domain, Expression text, Expression Name, Owner, and Expression profiling Level.

To add an Expression

1. Click Add Expression at the top of the Profiler screen.

2. Select a Domain from the Domain dropdown.
    - Domains are used by Profiling jobs to determine the masking Algorithm to apply to your sensitive data. When an Expression is matched, the Profiling job will associate the specified Domain to the sensitive data. The Masking Engine comes out of the box with over 30 pre-defined Domains. Domains can be added, edited, and deleted from the Settings Domains screen.
3. Enter the following information for the Expression:
    - Expression Name— The name used to select this expression as part of a Profiler Set.
    - Expression Text— The regular expression used to identify the sensitive data.
4. Select an Expression Level for the Expression:
    - Column Level— To identify sensitive data based on column names.
    - Data Level— To identify sensitive data based on data values, not column names.
5. When you are finished, click Save.

To edit a saved Expression, click the Edit icon to the right of the Expression.

To delete an Expression

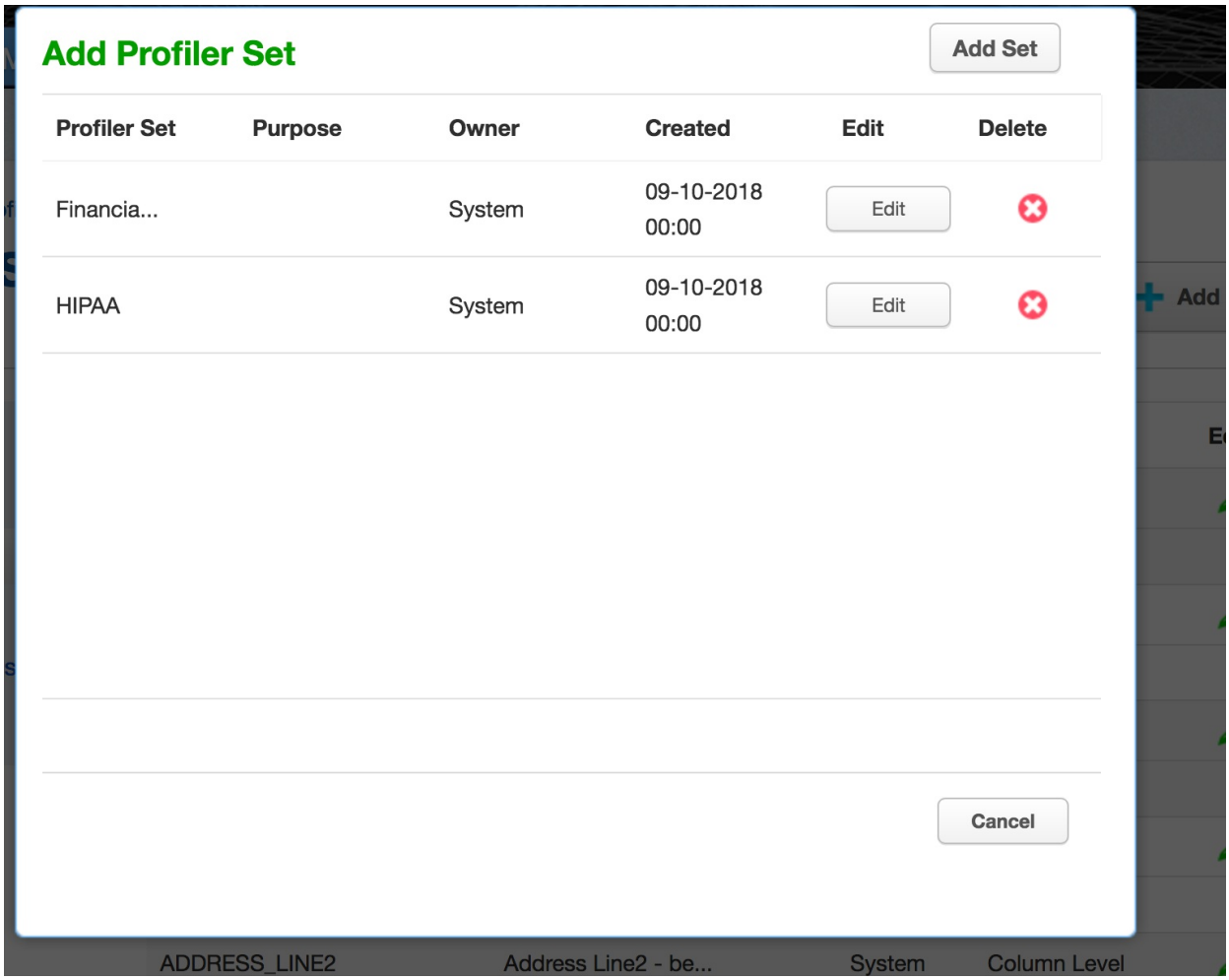
Click the Delete icon to the far right of the name.

Profiler Sets

Profiling jobs use Profiler Sets to determine the set of Expressions to use in identifying sensitive data in an Inventory. A Profiler Set is a grouping of Expressions for a particular purpose. For instance, First Name, Last Name, Address, Credit Card, SSN, and Bank Account Number Expressions could constitute a Financial Profiler Set.

The Masking Engine comes with two predefined Profiler Sets: Financial and Healthcare vertical. A Delphix Masking Engine administrator (a user with the appropriate role privileges) can create/add/update/delete these Profiler Sets.

If you want to edit or add a Profiler set, click Profiler Set at the top of the Profiler Settings screen. The Profiler Set dialog appears, listing the Profiler Sets along with their Purpose, Owner, and Date Created.



To add a Profiler Set

1. Click Add Set at the top of dialog.
2. Enter a Profiler Set Name.
3. Optionally, enter a Purpose for this Profiler Set.
4. Enter or select which Expressions to include in this set.
5. When you are finished, cick Submit.

To edit an existing Profiler Set, click the Edit icon to the right of the Profiler Set name.

To delete a Profiler Set

Click the Delete icon to the right of the Profiler Set name.

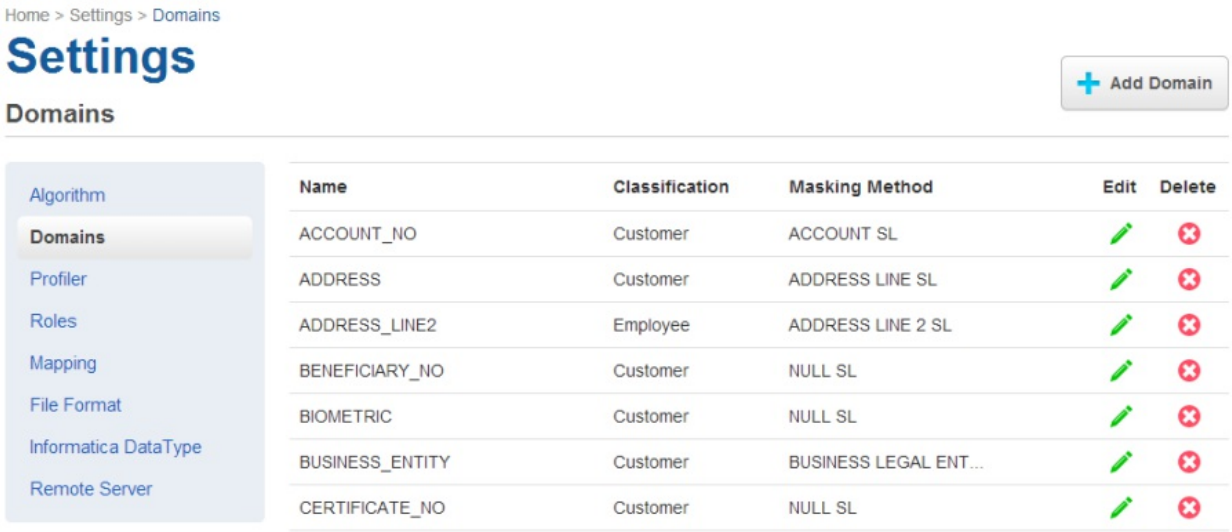
Managing Domains

This section describes how you can create and manage your domains.

Domains specify certain data to be masked with a certain algorithm. From the Settings tab, if you click Domains to the left, the list of domains will be displayed. From here, you can add, edit, or delete domains.

Delphix Agile Data Masking includes several default domains and algorithms. These appear the first time you display the Masking Settings tab. Each domain has a classification and masking method assigned to it. You might choose to assign a different algorithm to a domain, but each domain name is unique and can only be associated with one algorithm. If you create additional algorithms, they will appear in the Algorithms drop-down menu. Because each algorithm you use must have a unique domain, you must add a domain (or reassign an existing domain) to use any other algorithms.

The Domains tab is where you define domains, along with their classification and the default Masking Algorithm.



##Adding a New Domain

1. At the top of the Domains tab, click Add Domain.
2. Enter the new Domain Name. The domain name you specify will appear as a menu option on the Inventory screen elsewhere in the Delphix Masking Engine. Domain names must be unique.
3. Select the Classification (informational only). For example, customer-facing data, employee data, or company data.
4. Select a default Masking Algorithm for the new domain.

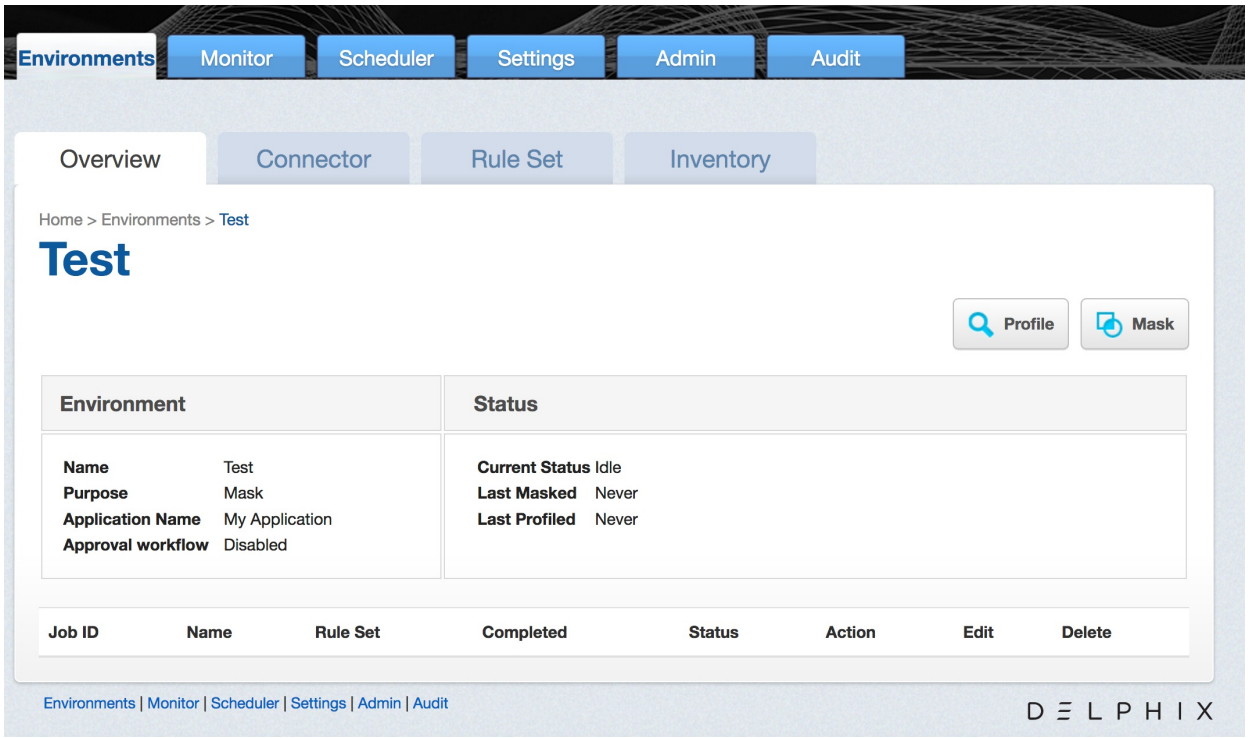
5. Click Save.
   To delete any domain, click the Delete icon to the far right of the domain name.

Creating A Profiling Job

This section describes how users can create a Profiling job. You can create Profiling jobs for databases, copybooks, delimited files, fixed-width, and Excel files.

The Profiler assigns each sensitive data element to a domain, with each domain having a default masking algorithm. Then, in the inventory, masking algorithms can be manually updated as needed to establish the masking rulesets for your data sources.

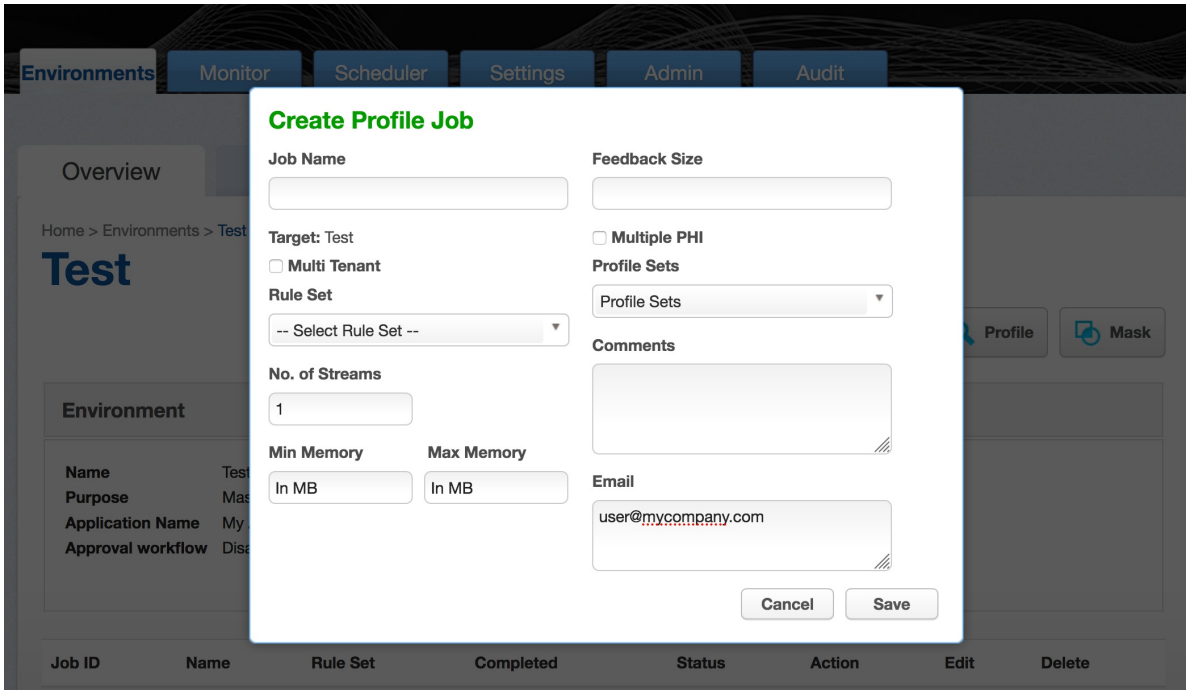
Profiling Jobs are grouped within environments on the Environment Overview page along with all masking jobs. In order to navigate to the Overview screen, click on an environment and the Overview tab should automatically display.



Creating a New Profiling Job

To create a new Profiling job:

1. Click the Profile button on the upper side of the page.
2. The Create Profiling Job window appears.



3. You will be prompted for the following information:

    - Job Name — A free-form name for the job you are creating. Must be unique.

    - Multi Tenant — Check the box if the job is for a multi-tenant database. This option allows existing rulesets to be re-used to mask identical schemas via different connectors. The connector is selected at job execution time.

    - Rule Set — Select the rule set that this job will profile.

    - No. of Streams — The number of parallel streams to use when running the jobs. For example, you can select two streams to profile two tables in the ruleset concurrently in the job instead of one table at a time.

    - Min Memory (MB) — (optional) Minimum amount of memory to allocate for the job, in megabytes.

    - Max Memory (MB) — (optional) Maximum amount of memory to allocate for the job, in megabytes.

    - Feedback Size — (optional) The number of rows to process before writing a message to the logs. Set this parameter to the appropriate level of detail required for monitoring your

job. For example, if you set this number significantly higher than the actual number of rows in a job, the progress for that job will only show 0 or 100%.

- Multiple PHI - Check the box if the job should run all Profile Expressions against the result set instead of finding the first matching Profile Expression. With this option, the Profiler report will indicate all matching Profile expressions, and if multiple Profile Expressions match, will assign the default Multiple PHI masking algorithm.

- Profile Sets — The name of the Profile Set to use.
  A Profile Set is a set of Profile Expressions (for example, a set of financial expressions). (See Delphix Administrator's Guide.)

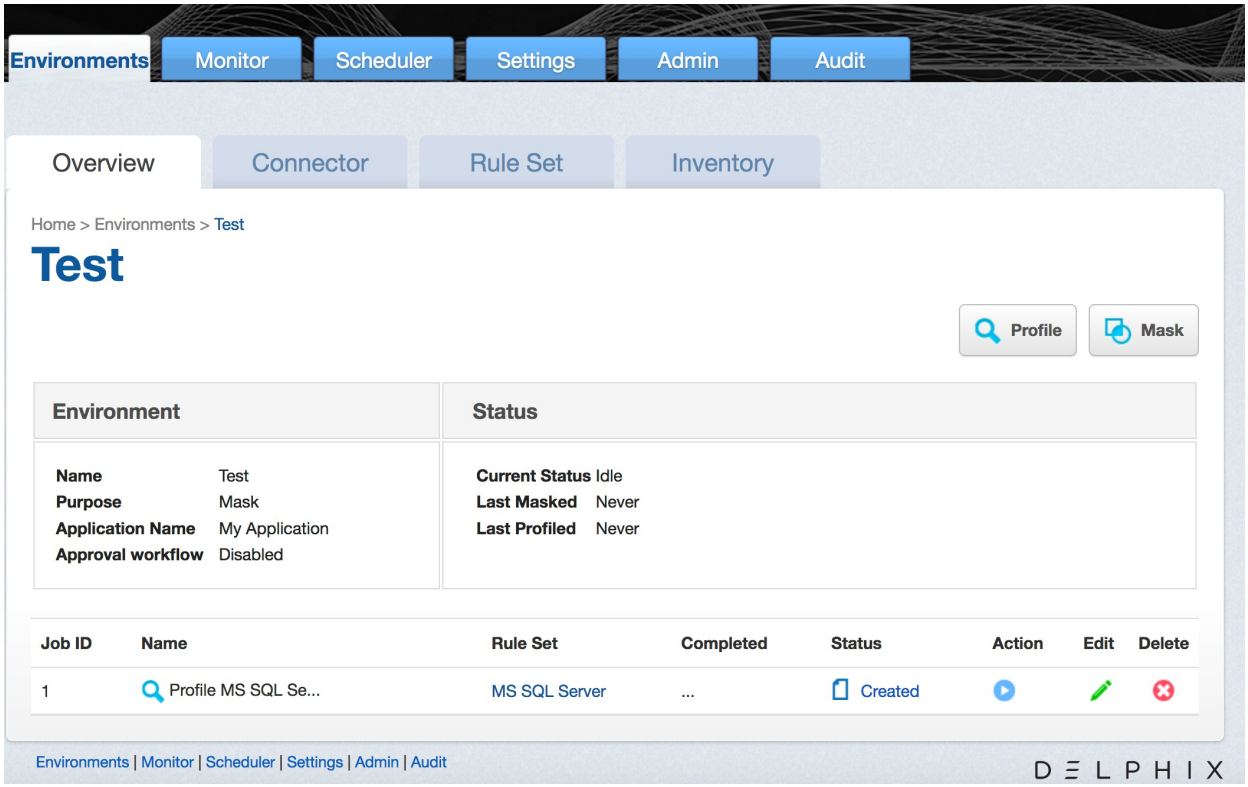
- Comments — (optional) Add comments related to this job.

- Email — (optional) Add e-mail address(es) to which to send status messages. Separate addresses with a comma (,).

4. When you are finished, click Save.

Running A Profiling Job

This section describes how users can run a profiling job from the Environment Overview screen.



To run or rerun a job from the Environment Overview screen:

- Click the Run icon (play icon) in the Action column for the desired job.
- The Run icon changes to a Stop icon while the job is running.
- When the job is complete, the Status changes.

To stop a running job from the Environment Overview screen:

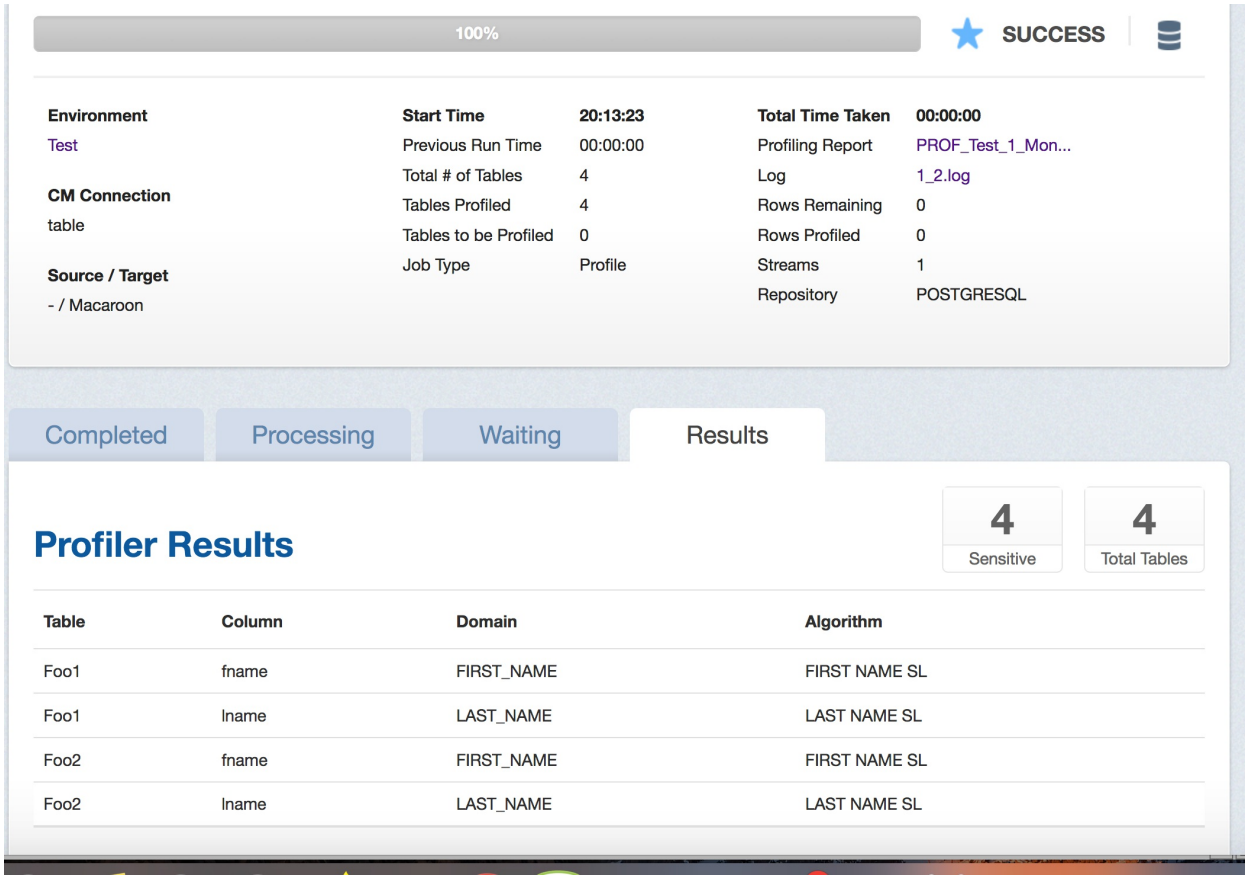
1. Locate the job you want to stop.
2. In the job's Action column, click the Stop icon.
3. A popup appears asking, "Are you sure you want to stop job?" Click OK.

When the job has been stopped, its status changes.

Reporting Profiling Results

This section describes the different ways of sharing/exploring the results of a Profiling job.

After a Job has been started from the Environment Overview screen, clicking on the Job Name will result in the display of the Profiling job from the Monitor tab. Clicking on the Results tab in the middle of the screen after the job has completed will display the sensitive data findings on a table-column by table-column or file-field by file-field basis.



To retrieve a PDF report of the Results tab, click on the Profiling Report link near the top of the page.

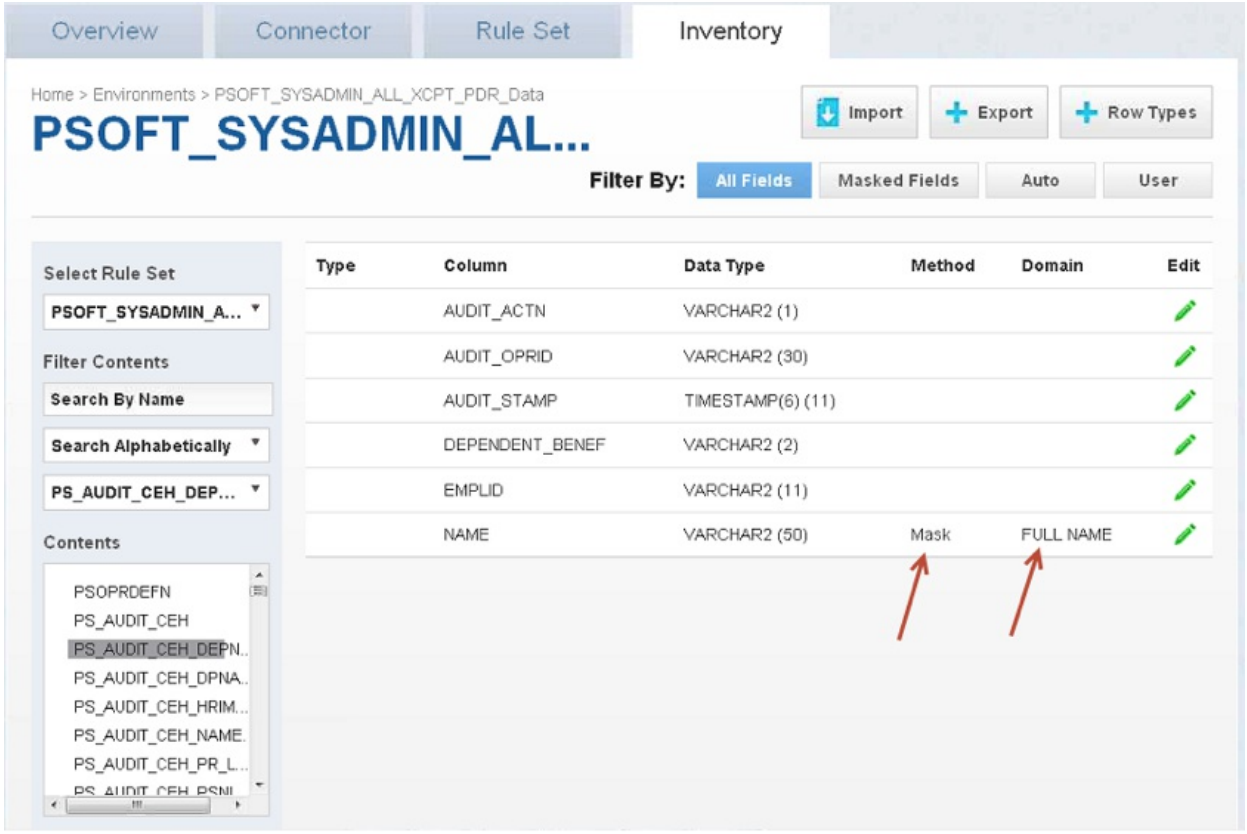
DELPHIX

Profiling Report

Job Profile		Job Status	
Name	MSSQLServer	Current Status :	Succeeded
Type :	Profiling	Start Time :	09/10/18 21:03
Environment :	Test	End Time :	09/10/18 21:03

Schema	Table	Column	Domain	Algorithm
dbo	Foo1	fname	FIRST_NAME	FIRST NAME SL
dbo	Foo1	lname	LAST_NAME	LAST NAME SL
dbo	Foo2	fname	FIRST_NAME	FIRST NAME SL
dbo	Foo2	lname	LAST_NAME	LAST NAME SL

Alternatively, after a job completes successfully, the profiling results can be displayed through the Inventory screen by examining the assigned Domain and masking algorithm Methods for tables/files in the Rule Set.



To get a spreadsheet capturing the Profiling results for the inventory, click on Export near the top of the page and a CSV file will be created.

MSSQL_Server_10917744884

Environment Name	Rule Set	Table Name	Type	Parent Column Name	Column Name	Data Type	Domain	Algorithm	Is Masked
Test	MSSQL Server	TEST4TABLE	-	-	TT1_COL1_DOT	varchar (100)	-	-	false
Test	MSSQL Server	TEST4TABLE	-	-	TT1_COL2_DOT	varchar (100)	-	-	false
Test	MSSQL Server	TEST4TABLE	-	-	ID1_DOT	int (0)	-	-	false
Test	MSSQL Server	Foo	-	-	IDD	int (0)	-	-	false
Test	MSSQL Server	Foo1	-	-	lname	varchar (50)	LAST_NAME	LAST NAME SL	true
Test	MSSQL Server	Foo1	-	-	fname	varchar (50)	FIRST_NAME	FIRST NAME SL	true
Test	MSSQL Server	Foo1	PK ID IX	-	idd	int (0)	-	-	false
Test	MSSQL Server	Foo2	PK ID IX	-	idd	int (0)	-	-	false
Test	MSSQL Server	Foo2	-	-	fname	varchar (50)	FIRST_NAME	FIRST NAME SL	true
Test	MSSQL Server	Foo2	-	-	lname	varchar (50)	LAST_NAME	LAST NAME SL	true

The spreadsheet can then be shared and manually modified to correct the sensitive data findings by:

1. Changing the Is Masked, Algorithm, and/or Domains fields for the respective Table/Column or File/Field in the CSV file accordingly.
2. Importing the modified spreadsheet by clicking on Import near the top of the Inventory screen and specifying the modified CSV file name.

Out Of The Box Algorithm Frameworks

This section describes the different algorithm frameworks (Secure Lookup, Segment Mapping, etc) that are available.

Secure Lookup Algorithm Framework

Secure lookup is the most commonly used type of algorithm. It is easy to generate and works with different languages. When this algorithm replaces real, sensitive data with fictional data, it is possible that it will create repeating data patterns, known as “collisions.” For example, the names “Tom” and “Peter” could both be masked as “Matt.” Because names and addresses naturally recur in real data, this mimics an actual data set. However, if you want the masking engine to mask all data into unique outputs, you should use segment mapping.

Sement Mapping Algorithm Framework

Segment mapping algorithms produce no overlaps or repetitions in the masked data. They let you create unique masked values by dividing a target value into separate segments and masking each segment individually.

You can mask up to a maximum of 36 values using segment mapping. You might use this method if you need columns with unique values, such as Social Security Numbers, primary key columns, or foreign key columns. When using segment mapping algorithms for primary and foreign keys, in order to make sure they match, you must use the same segment mapping algorithm for each. You can set the algorithm to produce alphanumeric results (letters and numbers) or only numbers.

With segment mapping, you can set the algorithm to ignore specific characters. For example, you can choose to ignore dashes [-] so that the same Social Security Number will be identified no matter how it is formatted. You can also preserve certain values. For example, to increase the randomness of masked values, you can preserve a single number such as 5 wherever it occurs. Or if you want to leave some information unmasked, such as the last four digits of Social Security numbers, you can preserve that information.

Segment Mapping Example

Perhaps you have an account number for which you need to create a segment mapping algorithm. You can separate the account number into segments, preserving the first two-character segment, replacing a segment with a specific value, and preserving a hyphen. The following is a sample value for this account number:

NM831026-04

Where:

- NM is a plan code number that you want to preserve, always a two-character alphanumeric code.
- 831026 is the uniquely identifiable account number. To ensure that you do not inadvertently create actual account numbers, you can replace the first two digits with a sequence that never appears in your account numbers in that location. (For example, you can replace the first two digits with 98 because 98 is never used as the first two digits of an account number.) To do that, you want to split these six digits into two segments.
- -04 is a location code. You want to preserve the hyphen and you can replace the two digits with a number within a range (in this case, a range of 1 to 77).

Mapping Algorithm Framework

A mapping algorithm allows you to state what values will replace the original data. It sequentially maps original data values to masked values that are pre-populated to a lookup table through the Masking Engine user interface. There will be no collisions in the masked data, because it always matches the same input to the same output. For example “David” will always become “Ragu,” and “Melissa” will always become “Jasmine.” The algorithm checks whether an input has already been mapped; if so, the algorithm changes the data to its designated output.

You can use a mapping algorithm on any set of values, of any length, but you must know how many values you plan to mask. You must supply AT MINIMUM the same number of values as the number of unique values you are

masking; more is acceptable. For example, if there are 10,000 unique values in the column you are masking you must give the mapping algorithm AT LEAST 10,000 values.

!!! info
When you use a mapping algorithm, you cannot mask more than one table at a time. You must mask tables serially.

Binary Lookup Algorithm Framework

A Binary Lookup Algorithm is much like the Secure Lookup Algorithm, but is used when entire files are stored in a specific column. This algorithm replaces objects that appear in object columns. For example, if a bank has an object column that stores images of checks, you can use a binary lookup algorithm to mask those images. The Delphix Engine cannot change data within images themselves, such as the names on X-rays or driver's licenses. However, you can replace all such images with a new, fictional image. This fictional image is provided by the owner of the original data.

Tokenization Algorithm Framework

A tokenization algorithm is the only type of algorithm that allows you to reverse its masking. For example, you can use a tokenization algorithm to mask data before you send it to an external vendor for analysis. The vendor can then identify accounts that need attention without having any access to the original, sensitive data. Once you have the vendor's feedback, you can reverse the masking and take action on the appropriate accounts.

Like mapping, a tokenization algorithm creates a unique token for each input such as "David" or "Melissa." The actual data (for example, names and addresses) are converted into tokens that have similar properties to the original data – such as text and length – but no longer convey any meaning. The Delphix Masking Engine stores both the token and the original so that you can reverse masking later.

Min Max Algorithm Framework

The Delphix Masking Engine provides a "Min Max Algorithm" to normalize data within a range – for example, 10 to 400. Values that are extremely high or low in certain categories allow viewers to infer someone's identity, even if their name has been masked. For example, a salary of $1 suggests a company's CEO, and some age ranges suggest higher insurance risk. You can use a min max algorithm to move all values of this kind into the midrange. This algorithm allows you to make sure that all the values in the database are within a specified range.

If the Out of range Replacement Values checkbox is selected, a default value is used when the input cannot be evaluated.

Data Cleansing Algorithm Framework

A data cleansing algorithm does not perform any masking. Instead, it standardizes varied spellings, misspellings, and abbreviations for the same name. For example, "Ariz," "Az," and "Arizona" can all be cleansed to "AZ." Use this algorithm if the target data needs to be in a standard format prior to masking.

Free Text Algorithm Framework

A free text redaction algorithm helps you remove sensitive data that appears in free-text columns such as "Notes." This type of algorithm requires some expertise to use, because you must set it to recognize sensitive data within a block of text.

One challenge is that individual words might not be sensitive on their own, but together they can be. The algorithm uses profiler sets to determine what information it needs to mask. You can decide which expressions the algorithm uses to search for material such as addresses. For example, you can set the algorithm to look for "St," "Cir," "Blvd," and other words that suggest an address. You can also use pattern matching to identify potentially sensitive information. For example, a number that takes the form 123-45-6789 is likely to be a Social Security Number.

You can use a free text redaction algorithm to show or hide information by displaying either a "black list" or a "white list."

Blacklist – Designated material will be redacted (removed). For

example, you can set a blacklist to hide patient names and addresses. The blacklist feature will match the data in the lookup file to the input file.

Whitelist – ONLY designated material will be visible. For example, if a drug company wants to assess how often a particular drug is being prescribed, you can use a white list so that only the name of the drug will appear in the notes. The whitelist feature enables you to mask data using both the lookup file and a profile set.

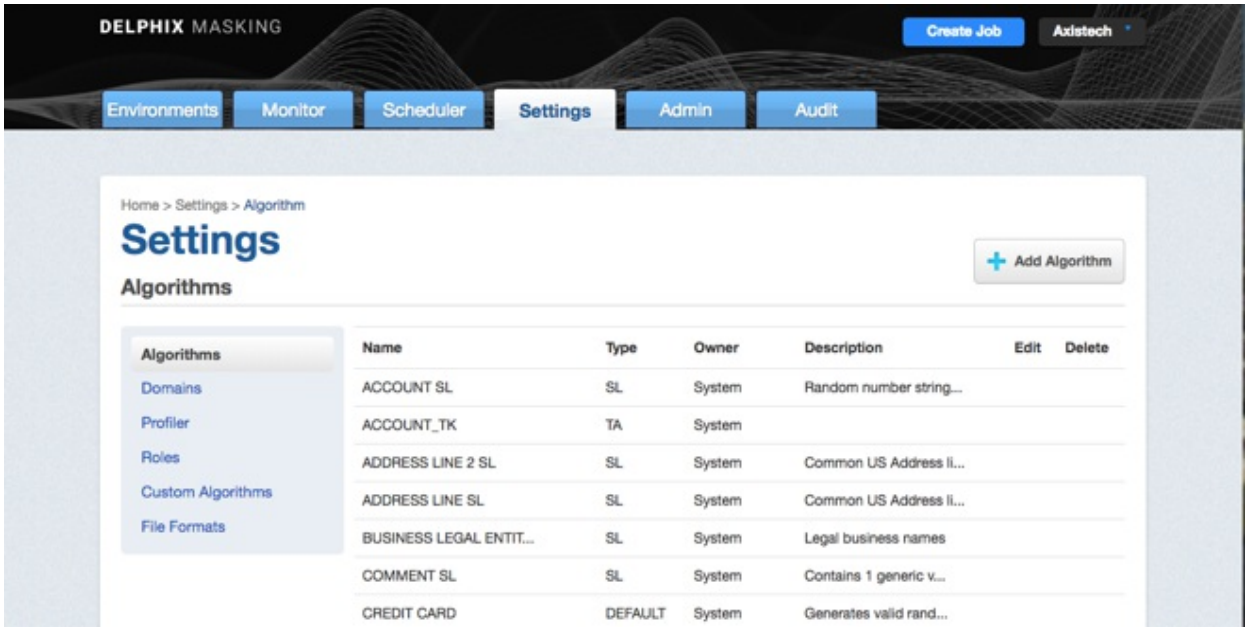
For either option, a list of words can be imported from an external text file or alternatively, you can use Profiler Sets to match words based on regular expressions, defined within Profiler Expressions. You can also specify the redaction value that will replace the masked words. Regular expressions defined using Profiler Sets will match individual words within the input text, rather than phrases.

Configuring Your Own Algorithms

This section describes how users can configure their own algorithms using Delphix's built in algorithm frameworks.

Algorithm Settings

The Algorithm tab displays algorithm Names along with Type and Description. This is where you add (or create) new algorithms. The default algorithms and any algorithms you have defined appear on this tab. All algorithm values are stored encrypted. These values are only decrypted during the masking process.



Creating New Algorithms

If none of the default algorithms meet your needs, you might want to create a new algorithm.

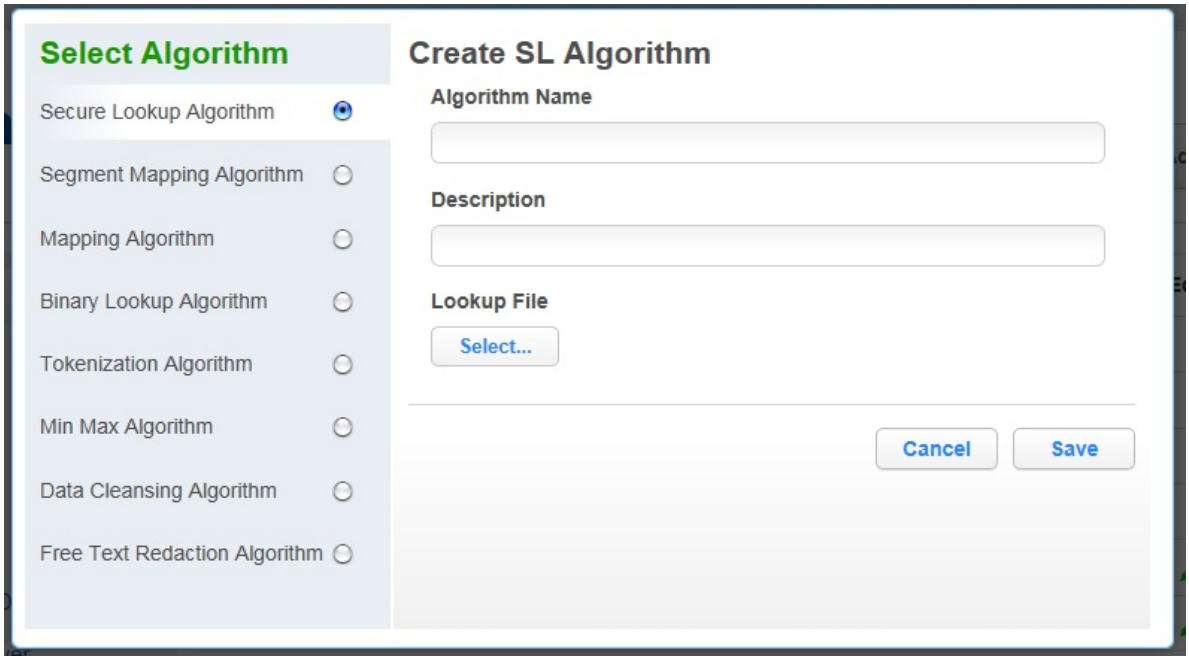
Algorithm Frameworks give you the ability to quickly and easily define the algorithms you want, directly on the Settings page. Then, you can immediately propagate them. Anyone in your organization who has the Delphix Masking Engine can then access the information.

Administrators can update system-defined algorithms. User-defined

algorithms can be accessed by all users and updated by the owner/user who created the algorithm.

To add an algorithm:

1. In the upper right-hand corner of the Algorithm settings tab, click Add Algorithm.



2. Select an algorithm type.

3. Complete the form to the right to name and describe your new algorithm.

4. Click Save.

Choosing an Algorithm Framework

See Out Of The Box Secure Methods/Algorithms for detailed description on each Algorithm Framework. The algorithm framework you choose will depend on the format of the data & your internal data security guidelines.

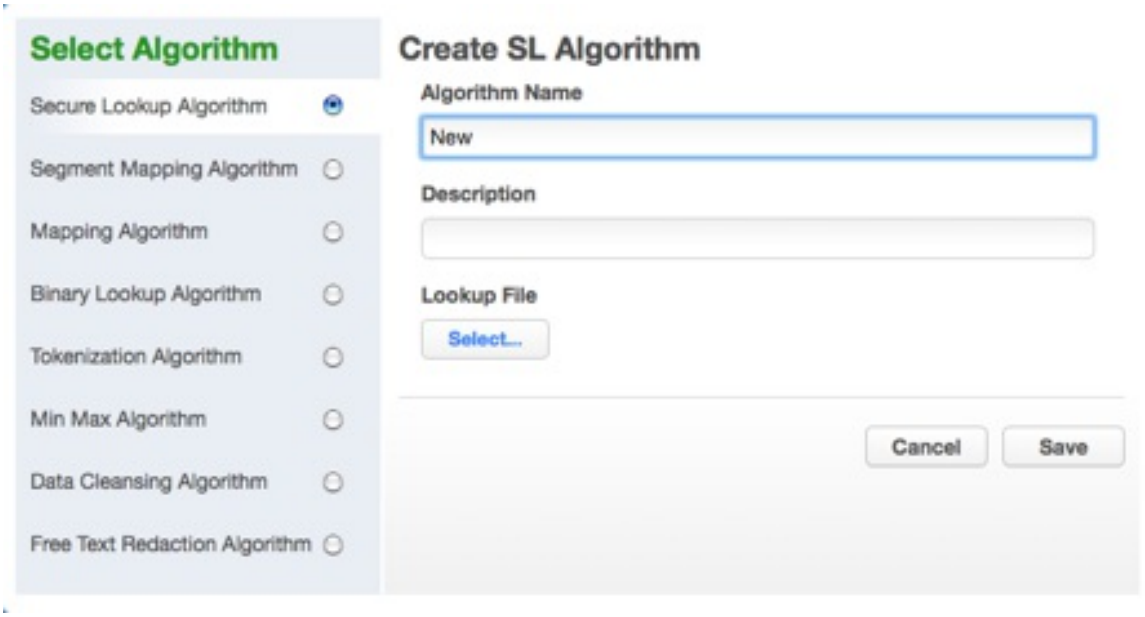
Secure Lookup Algorithm Framework

To add a secure lookup algorithm:

1. In the upper right-hand corner of the Algorithm tab, click

Add Algorithm.

2. Choose Secure Lookup Algorithm. The Create SL Algorithm pane appears.



3. Enter a Algorithm Name.

   !!! info
   This MUST be unique.

4. Enter a Description.

5. Specify a Lookup File.

   This file is a single list of values. It does not require a header.
   Make sure there are no spaces or returns at the end of the last line in the file. The following is sample file content:

```
Smallville
Clarkville
Farmville
Townville
Cityname
Citytown
Towneaster
```

6. When you are finished, click Save.

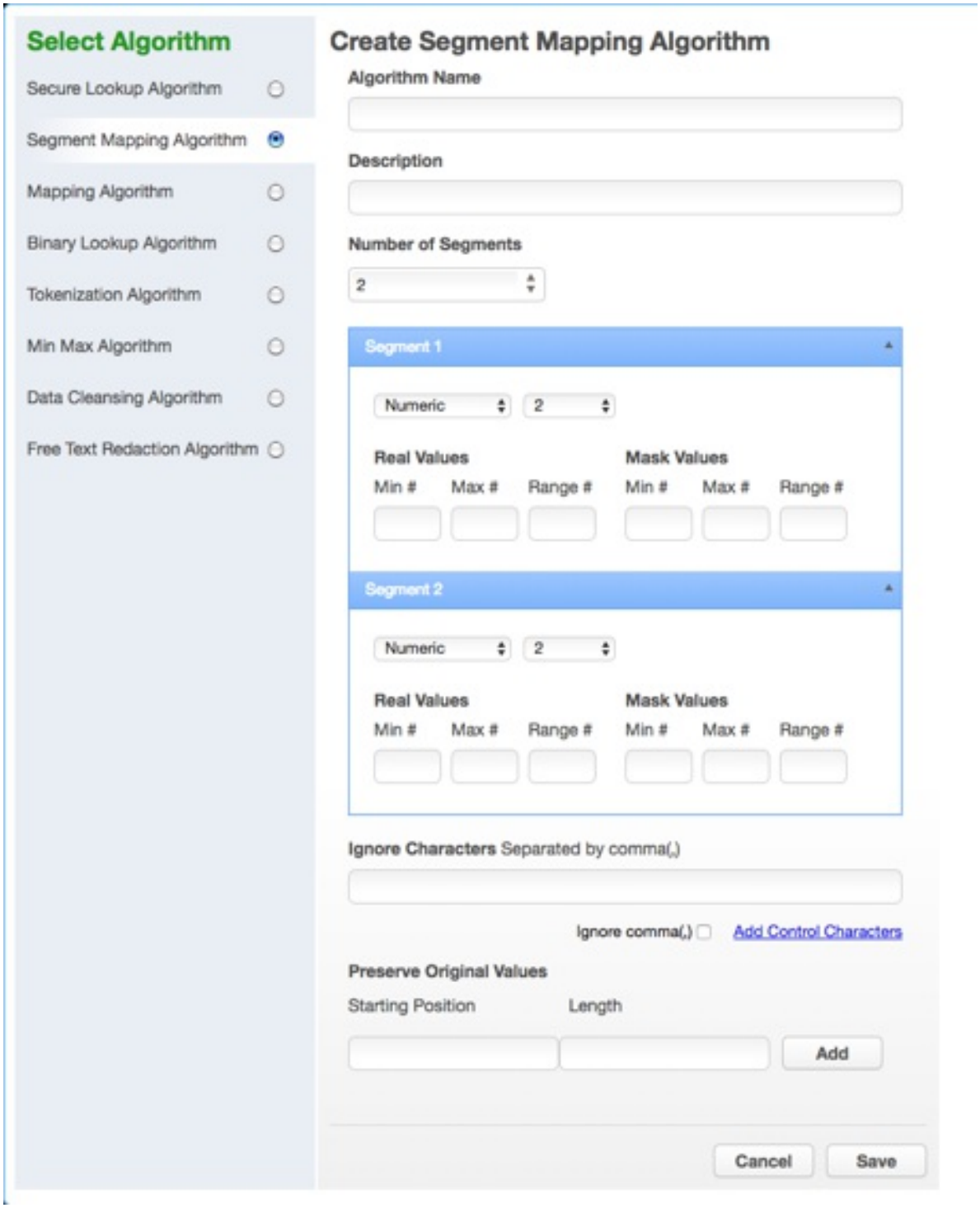
7. Before you can use the algorithm in a profiling job, you must add it to a domain.

!!! info
The masking engine supports lookup files saved in ASCII or UTF-8 format only. If the lookup file contains foreign alphabet characters, the file must be saved in UTF-8 format with no BOM (Byte Order Marker) for Masking Engine to read the Unicode text correctly. Some applications, e.g. Notepad on Windows, write a BOM (Byte Order Marker) at the beginning of Unicode files which irritates the masking engine and will lead to SQL update or insert errors when trying to run a masking job that applies a Secure Lookup algorithm that has been created based on a UTF-8 file that included a BOM.

Segmented Mapping Algorithm Framework

1. In the upper right-hand region of the Algorithm tab, click Add Algorithm.

2. Select Segment Mapping Algorithm. The Create Segment Mapping Algorithm pane appears.



3. Enter a Rule Name.

4. Enter a Description.

5. From the No. of Segment drop-down menu, select how many segments you want to mask.

   !!! note "NOTE"
   This number does NOT include the values you want to preserve.

The minimum number of segments is 2; the maximum is 9. A box appears for each segment.

6. For each segment, choose the Type of segment from the dropdown: Numeric or Alphanumeric.

   !!! info
   Numeric segments are masked as whole segments. Alphanumeric segments are masked by individual character.

7. For each segment, select its Length (number of characters) from the drop-down menu. The maximum is 4.

8. Optionally, for each segment, specify range values. You might need to specify range values to satisfy particular application requirements, for example. See details below.

9. Preserve Original Values by entering Starting position and length values. (Position starts at 1.) For example, to preserve the second, third, and fourth values, enter Starting position 2 and length 3.

   If you need additional value fields, click Add.

10. When you are finished, click Save.

11. Before you can use the algorithm in a profiling job, you must add it to a domain. If you are not using the Masking Engine Profiler to create your inventory, you do not need to associate the algorithm with a domain.

Specifying Range Values

You can specify ranges for Real Values and Mask Values. With Real Values ranges, you can specify all the possible real values to map to the ranges of masked values. Any values NOT listed in the Real Values ranges would then mask to themselves.

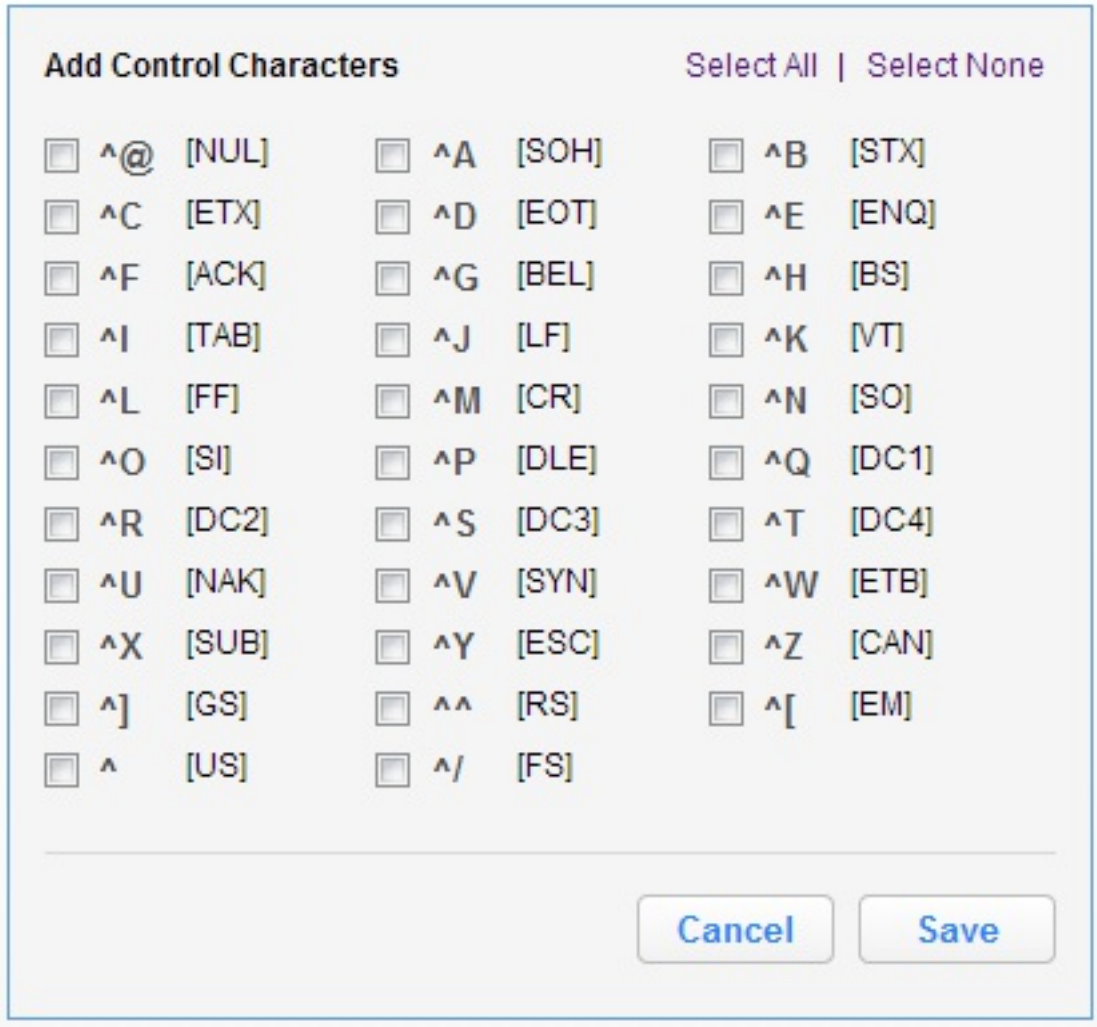
Specifying range values is optional. If you need unique values (for example, masking a unique key column), you MUST leave the range values blank. If you plan to certify your data, you must specify range values.

When determining a numeric or alphanumeric range, remember that a narrow range will likely generate duplicate values, which will cause your job to fail.

1. To ignore specific characters, enter one or more characters in the Ignore Character List box. Separate values with a comma.

2. To ignore the comma character (,), select the Ignore comma (,) check box.

3. To ignore control characters, select Add Control Characters. The Add Control Characters window appears.



4. Select the individual control characters that you would like to ignore, or choose Select All or Select None.

5. When you are finished, click Save.

6. You are returned to the Segment Mapping pane.

Numeric segment type

- Min# — A number; the first value in the range. Value can be 1 digit or up to the length of the segment. For example, for a 3-digit segment, you can specify 1, 2, or 3 digits. Acceptable characters: 0-9.

- Max# — A number; the last value in the range. Value should be the same length as the segment. For example, for a 3-digit segment, you should specify 3 digits. Acceptable characters: 0-9.

- Range# — A range of numbers; separate values in this field with a comma (,). Value should be the same length as the segment. For example, for a 3-digit segment, you should specify 3 digits. Acceptable characters:
  0-9.

!!! info
If you do not specify a range, the Masking Engine uses the full range. For example, for a 4-digit segment, the Masking Engine uses 0-9999.

Alphanumeric segment type

- Min# — A number from 0 to 9; the first value in the range.

- Max# — A number from 0 to 9; the last value in the range.

- MinChar — A letter from A to Z; the first value in the range.

- MaxChar — A letter from A to Z; the last value in the range.

- Range# — A range of alphanumeric characters; separate values in this field with a comma (,). Individual values can be a number from 0 to 9 or an uppercase letter from A to Z. (For example, B,C,J,K,Y,Z or AB,DE.)

!!! info
If you do not specify a range, the Masking Engine uses the full range (A-Z, 0-9). If you do not know the format of the input, leave the range fields empty. If you know the format of the input (for example, always alphanumeric followed by numeric), you can enter range values such as A2 and S9.

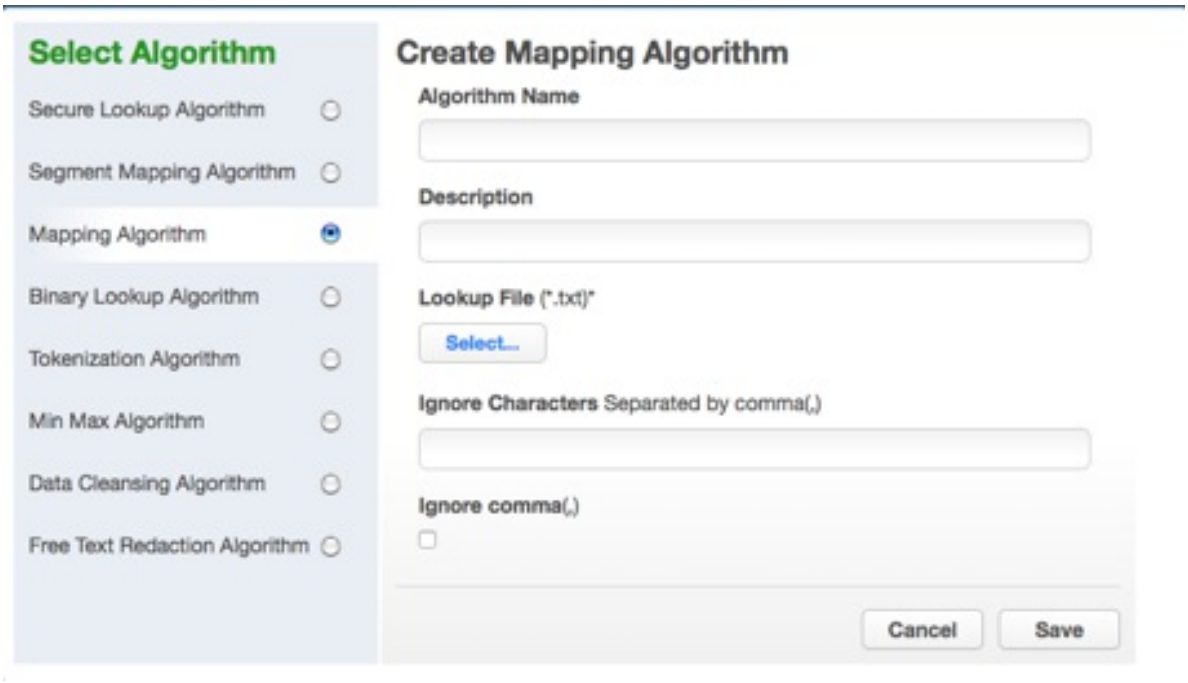
Mapping Algorithm Framework

To add a mapping algorithm:

1. In the upper right-hand corner of the Algorithm tab, click Add Algorithm.

2. Select Mapping Algorithm.

3. The Create Mapping Algorithm pane appears.



4. Enter a Rule Name. This name MUST be unique.

5. Enter a Description.

6. Specify a Lookup File.

7. The value file must have NO header. Make sure there are no spaces or returns at the end of the last line in the file. The following is sample file content. Notice that there is no header and only a list of values.

```
Smallville
Clarkville
Farmville
Townville
Cityname
```

```
Citytown
Towneaster
```

8. To ignore specific characters, enter one or more characters in the Ignore Character List box. Separate values with a comma.

9. To ignore the comma character (,), select the Ignore comma (,) check box.

10. When you are finished, click Save.

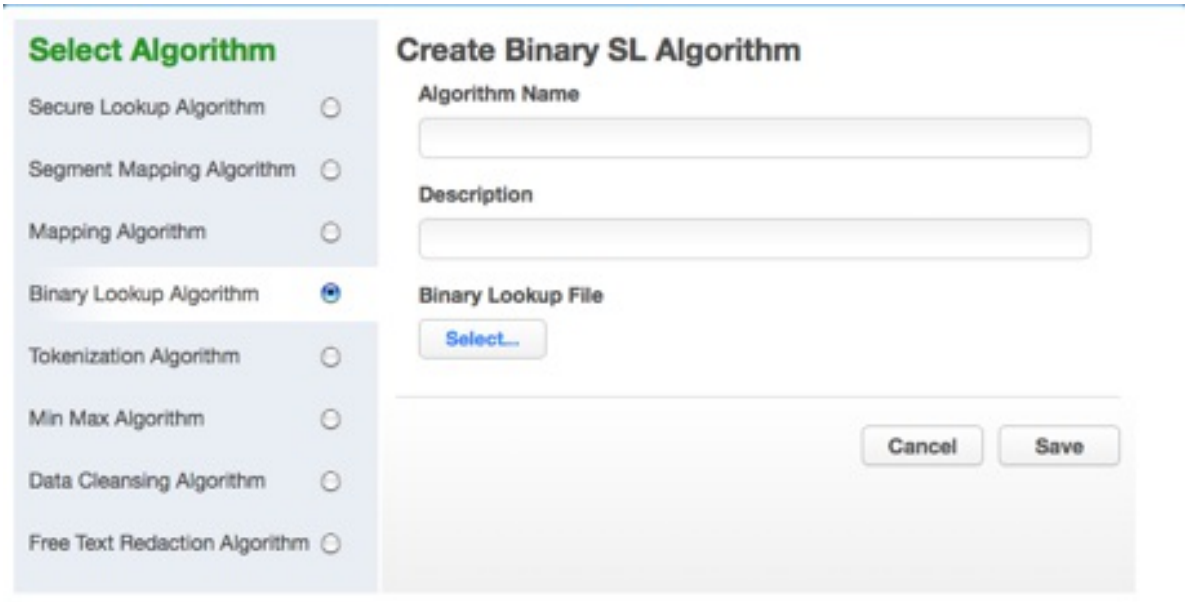
Before you can use the algorithm by specifying it in a profiling job, you must add it to a domain. If you are not using the Masking Engine Profiler to create your inventory, you do not need to associate the algorithm with a domain.

Masking Binary Lookup Algorithm Framework

To add a binary lookup algorithm:

1. At the top right of the Algorithm tab, click Add Algorithm.

2. Select Binary Lookup Algorithm. The Binary SL Rule pane appears.



3. Enter a Rule Name.

4. Enter a Description.

5. Select a Binary Lookup File on your filesystem.

6. Click Save.

Tokenization Algorithm Framework

To add a Tokenization algorithm:

1. Enter algorithm Name.

2. Enter a Description.

3. Click Save.

Once you have created an algorithm, you will need to associate it with a domain.

1. Navigate to the Home>Settings>Domains page and click Add Domain.

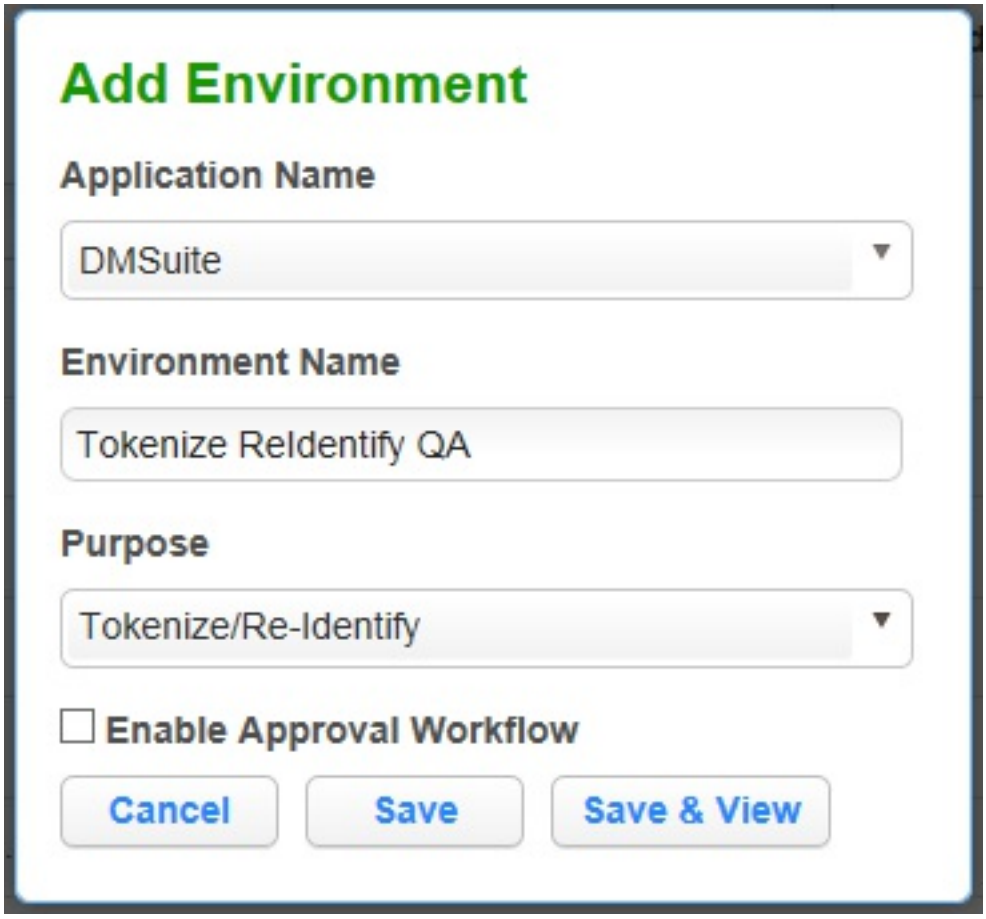
2. Enter a domain name.

3. From the Tokenization Algorithm Name drop-down menu, select your algorithm.

Next, create a Tokenization Environment:

1. On the home page, click Environments.

2. Click Add Environment.

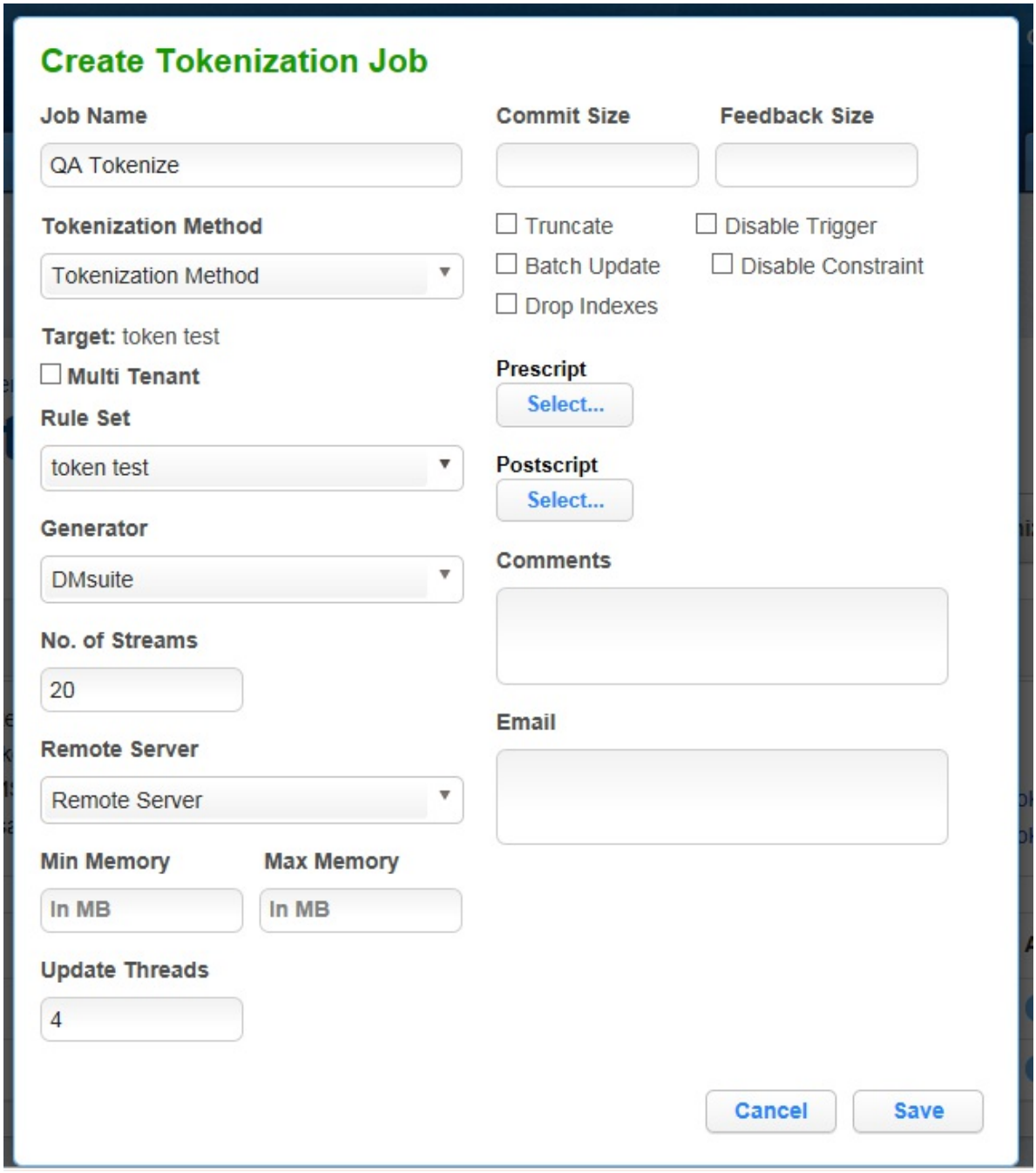


3. For Purpose, select Tokenize/Re-Identify.

4. Click
   Save.

   !!! info
   This environment will be used to re-identify your data when required.

5. Set up a Tokenize job using tokenization method. Execute the job.



Here is a snapshot of the data before and after Tokenization to give you an idea of what the it will look like.

Before Tokenization

```
ID,fname,address,ssn
1,Erasmus,245 Park Ave,123-45-6789
2,Ridley,1003 Stant Drive,123-45-6789
3,Jason,45 Omega Suites,123-45-6789
4,Waldeve,1 Pulitzer way,123-45-6789
5,Salathiel,245 park Ave,123-45-6789
```

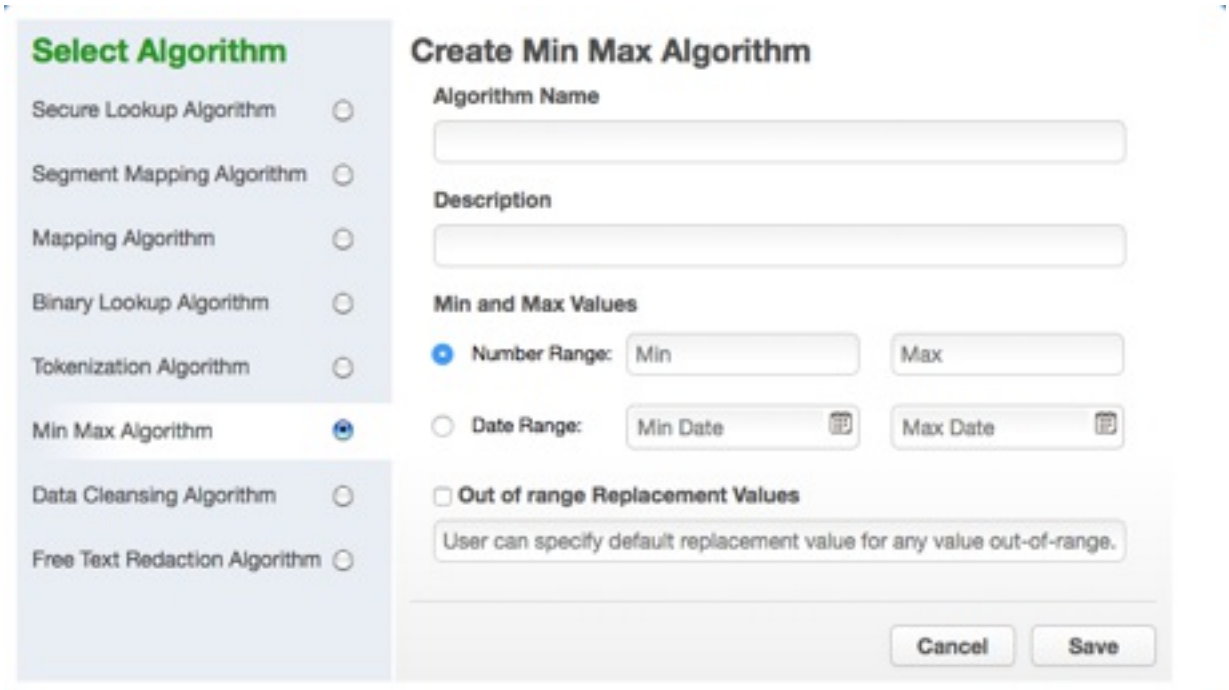
After Tokenization

```
ID,fname,address,ssn
1,Erasmus,L1kgrFFRzafOTUqfpZAmiC==,123-45-6789
2,Ridley,+7A16uqP1BSbaaL1f0T7lzqijNVHU38Z2fMMK0fX4+O=,123-45-6789
3,Jason,C4v5jrlmKEhKC3acnQKqEk==,123-45-6789
4,Waldeve,v89pB9b9QISxyYvs/agYUg==,123-45-6789
5,Salathiel,yrLNBhI8j40ld7y7dXRqwY==,123-45-6789
```

MIN Max Algorithm Framework

The Delphix Masking Engine provides a "Min Max Algorithm" to normalize data within a range – for example, 10 to 400. Values that are extremely high or low in certain categories allow viewers to infer someone's identity, even if their name has been masked. For example, a salary of $1 suggests a company's CEO, and some age ranges suggest higher insurance risk. You can use a min max algorithm to move all values of this kind into the midrange. This algorithm allows you to make sure that all the values in the database are within a specified range.

If the Out of range Replacement Values checkbox is selected, a default value is used when the input cannot be evaluated.



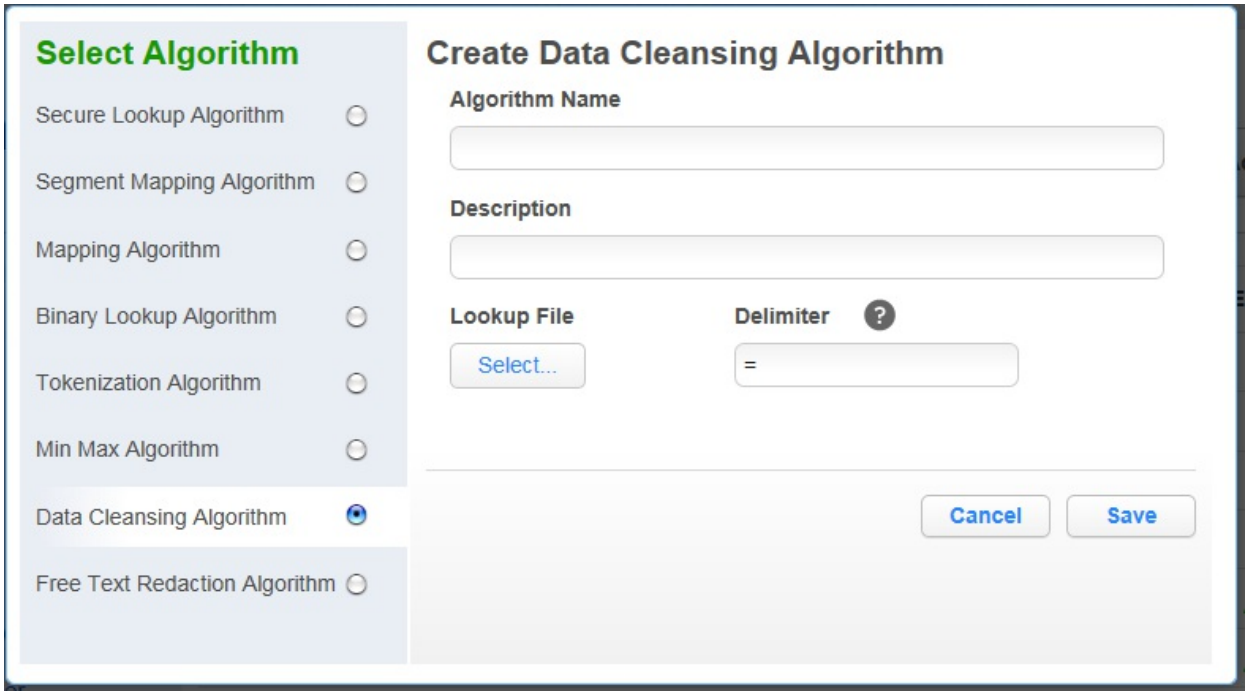
1. Enter the Algorithm Name.
2. Enter a Description.

3. Enter Min Value and Max Value.
4. Click Out of range Replacement Values.
5. Click Save.

Example: Age less than 18 years - enter Min Value 0 and Max Value 18.

Data Cleansing Algorithm Framework

A data cleansing algorithm does not perform any masking. Instead, it standardizes varied spellings, misspellings, and abbreviations for the same name. For example, “Ariz,” “Az,” and “Arizona” can all be cleansed to “AZ.” Use this algorithm if the target data needs to be in a standard format prior to masking.



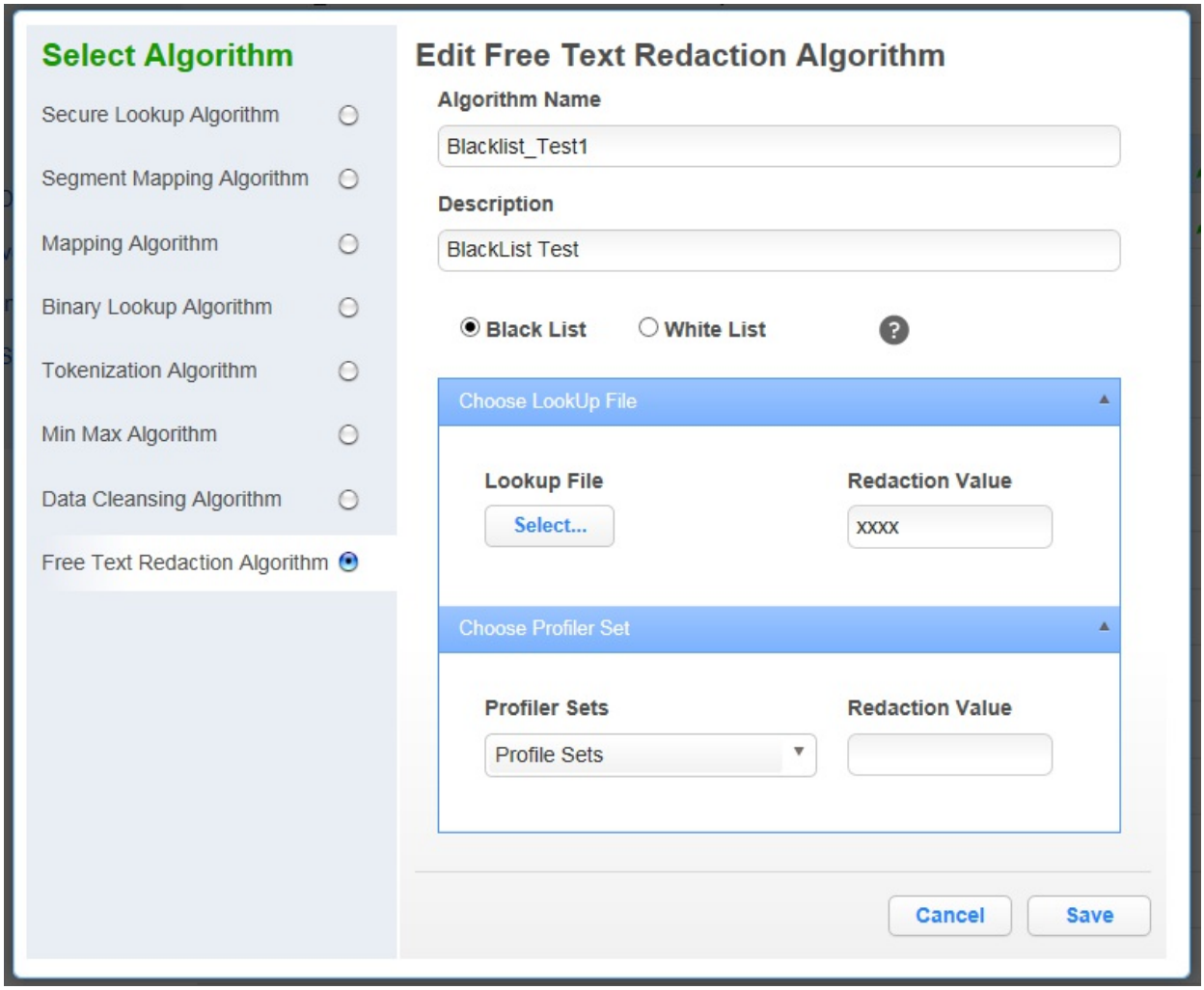
1. Enter an Algorithm Name.
2. Enter a Description.
3. Select Lookup File location.
4. Specify a Delimiter (key and value separator). The default delimiter is =. You can change this to match the lookup file.
5. Click Save.

Below is an example of a lookup input file. It does not require a header. Make sure there are no spaces or returns at the end of the last line in the file. The following is sample file content:

```
NYC=NY
NY City=NY
New York=NY
Manhattan=NY
```

Free Text Algorithm Framework

To add a free text redaction algorith:



1. Enter an Algorithm Name.
2. Enter a Description.
3. Select the Black List or White List radio button.

4. Select Lookup File and enter Redaction Value OR/AND

5. Select Profiler Sets from the drop-down menu and enter Redaction Value.

6. Click Save.

Free Text Redaction Example

1. Create Input File.

2. Create input file using notepad. Enter the following text:

```
The customer Bob Jones is satisfied with the terms of the sales
agreement. Please call to confirm at 718-223-7896.
```

3. Save file as txt.

4. Create lookup file.

   i. Create a lookup file.

   ii. Use notepad to create a txt file and save the file as a TXT. Be sure to hit return after each field. The lookup flat file contains the following data:

```
Bob
Jones
Agreement
```

Create an Algorithm

You will be prompted for the following information:

1. For Algorithm Name, enter Blacklist_Test1.

2. For Description, enter Blacklist Test.

3. Select the Black List radio button.

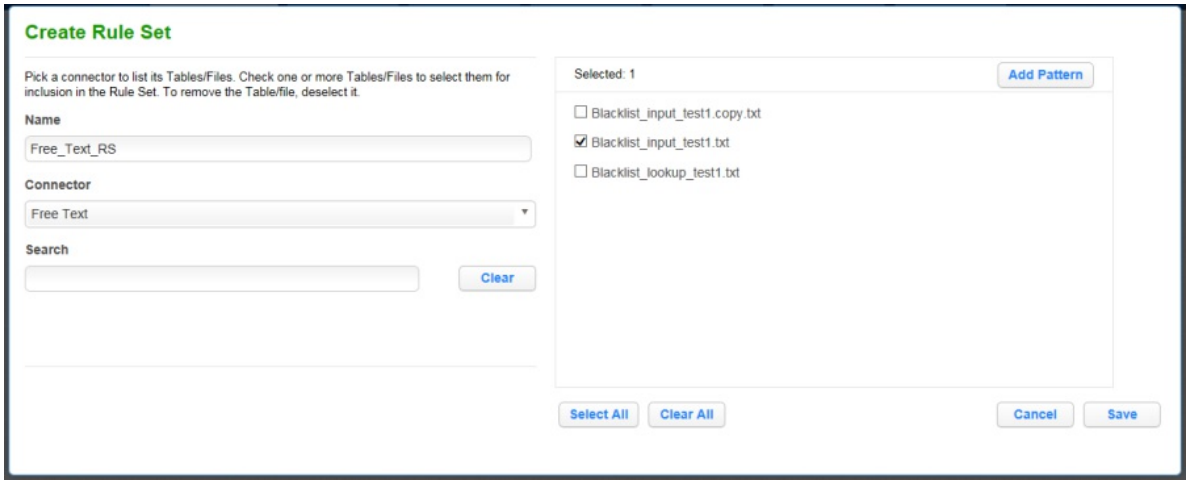
4. Select LookUp File.

5. Enter redaction value XXXX.

6. Click Save.

Create Rule Set

1. From the job page go to Rule Set and Click Create Rule Set.



2. For Rule Set Name, enter Free_ Text_RS.

3. From the Connector drop-down menu, select Free Text.

4. Select the Input File by clicking the box next to your input file

5. Click Save.

Create Masking Job

1. Use Free_Text Rule Set

2. Execute Masking job.

The results of the masking job will show the following:

```
The customer xxxx xxxx is satisfied with the terms
of the sales xxxx. Please call to confirm at 718-223-7896.
```

"Bob," "Jones," and "agreement" are redacted.

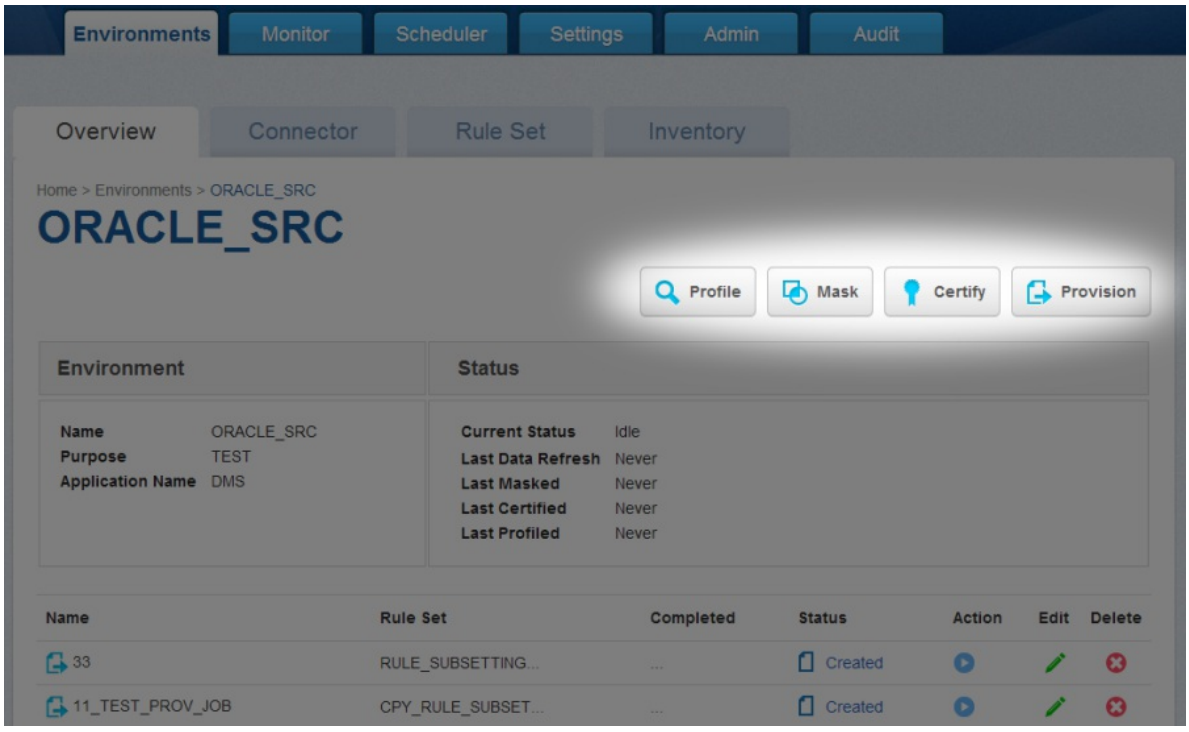
Creating Masking Job

This section describes how users can create a masking job.

Creating New Jobs

In the Environment Overview screen, select one of the jobs icons to create the corresponding job:

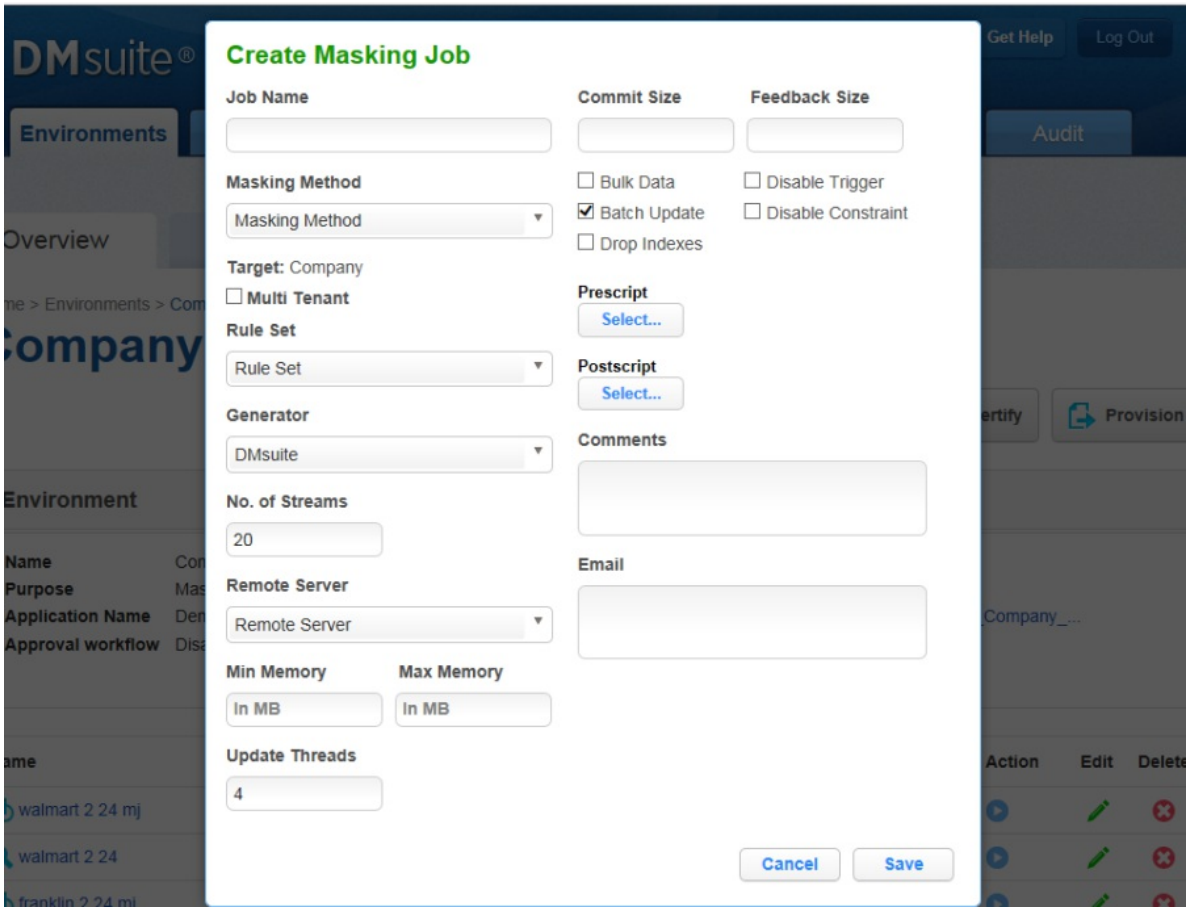
- Profile
- Mask



Creating a New Masking Job

To create a new masking job:

1. Click Mask. The Create Masking Job window appears.



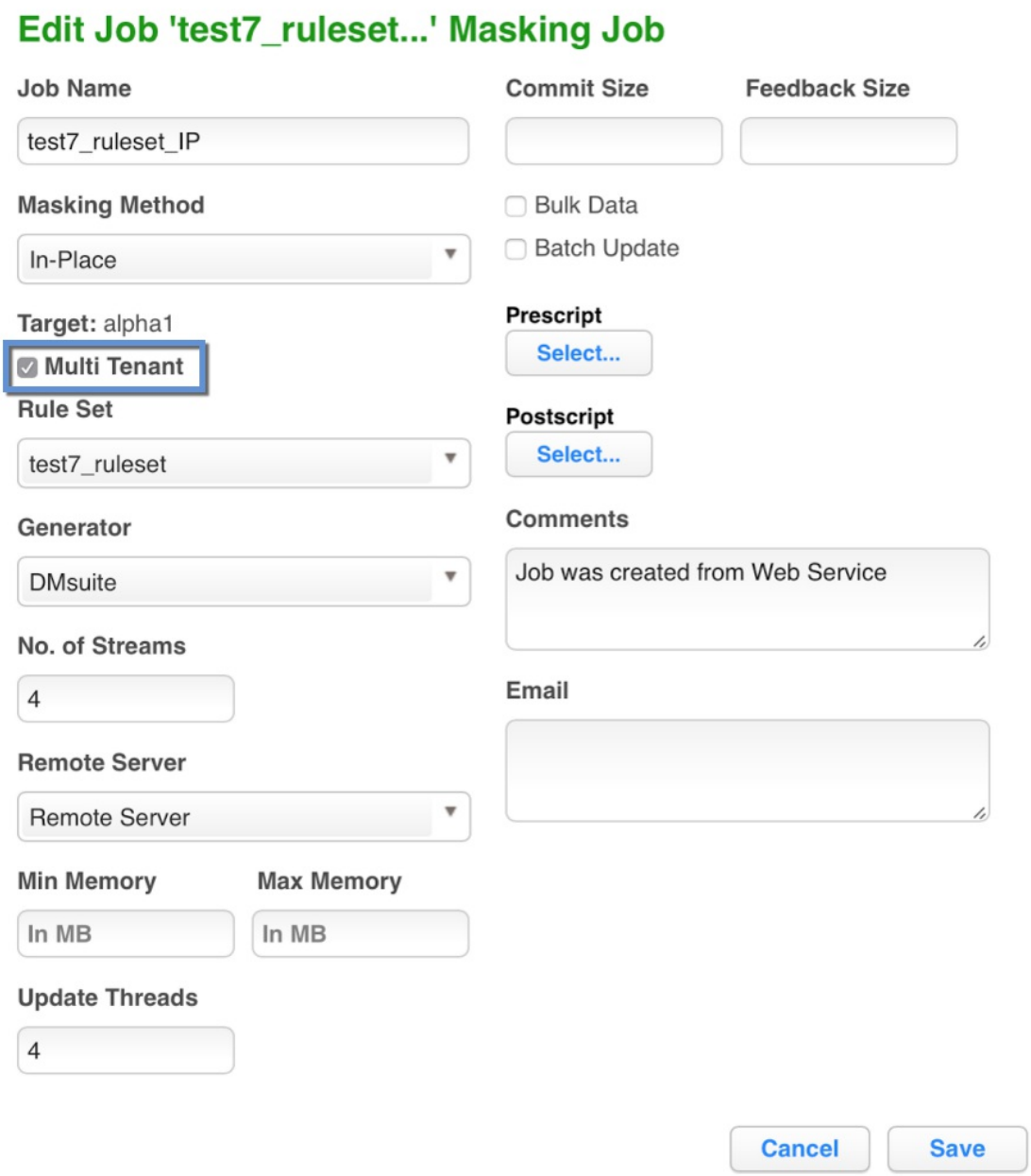
2. You will be prompted for the following information:

    i. Job Name — A free-form name for the job you are creating. Must be unique across the entire application.

    ii. Masking Method — Select either In-Place or On-The-Fly. For more information on masking type, see Mask Data. [Need to create this link]

    iii. Multi Tenant — Check box if the job is for a multi-tenant database.

    !!! info "INFO: Provisioning Masked VDBs."
    A job must be Multi Tenant to use it when creating a masked virtual database (VDB).



iv. Rule Set — Select a rule set that this job will execute against.

v. Generator — The default value is Delphix.

vi. Repository Folder name — The folder name in the repository where the objects should be imported.

vii. Parameter File Path — (optional) If checked, this tells Delphix to configure the sessions and workflows to use a parameter file that contains the source and target connection information.
If unchecked, the Delphix Engine will generate sessions/workflows that use the connector names as defined within the Delphix Engine,

which will require connections with the same names defined within the repository.

viii. Import Mapplet — (optional) if checked, this tells the Delphix Engine to import mapplets that are assigned to columns in the inventory along with the mappings/sessions/workflows. If unchecked, Delphix will not attempt to import any mapplets that are assigned in the inventory.

ix. Mask Method — Choose either of the following:

    a. No. of Streams—The number of parallel streams to use when running the jobs. For example, you can select two streams to run two tables in the Rule Set concurrently in the job instead of one table at a time.

    b. Import — When you click the Run icon, creates the mappings but does not execute the workflow. You later run the job.

    c. Import and Run — When you click the Run icon, creates the mappings and executes the workflow.

x. Remote Server — (optional) The remote server that will execute the jobs. This option lets you choose to execute jobs on a remote server, rather than on the local Delphix instance. Note: This is an optional feature for Delphix.

xi. Min Memory (MB) — (optional) Minimum amount of memory to allocate for the job, in megabytes.

xii. Max Memory (MB) — (optional) Maximum amount of memory to allocate for the job, in megabytes.

xiii. Update Threads — The number of update threads to run in parallel to update the target database.

    !!! info
    Multiple threads should not be used if the masking job contains any table without an index. Multi-threaded masking jobs can lead to deadlocks on the database engine.

```
Multiple threads can cause database engine deadlocks for databases using T-SQL If
```

xiv. Commit Size — (optional) The number of rows to process before issuing a commit to the database.

xv. Feedback Size — (optional) The number of rows to process before writing a message to the logs. Set this parameter to the appropriate level of detail required for monitoring your job. For example, if you set this number significantly higher than the actual number of rows in a job, the progress for that job will only show 0 or 100%.

xvi. Bulk Data — (optional) For In-Place masking only. The default is for this check box to be clear. If you are masking very large tables in-place and require performance improvements, check this box. Delphix will mask data to a flat file, and then use inserts instead of updates to bulk load the target table.

xvii. Disable Constraint — (optional) Whether to automatically disable database constraints. The default is for this check box to be clear and therefore not perform automatic disabling of constraints. For more information about database constraints, see Enabling and Disabling Database Constraints.

xviii. Batch Update — (optional) Enable or disable use of a batch for updates. A job's statements can either be executed individually, or can be put in a batch file and executed at once, which is faster.

xix. Disable Trigger — (optional) Whether to automatically disable database triggers. The default is for this check box to be clear and therefore not perform automatic disabling of triggers.

xx. Drop Index — (optional) Whether to automatically drop indexes on columns which are being masked and automatically re-create the index when the masking job is completed. The default is for this check box to be clear and therefore not perform automatic dropping of indexes.

xxi. Prescript — (optional) Specify the full pathname of a file that contains SQL statements to be run before the job starts, or click Browse to specify a file. If you are editing the job and a prescript file is already specified, you can click the Delete button to remove the file. (The Delete button only

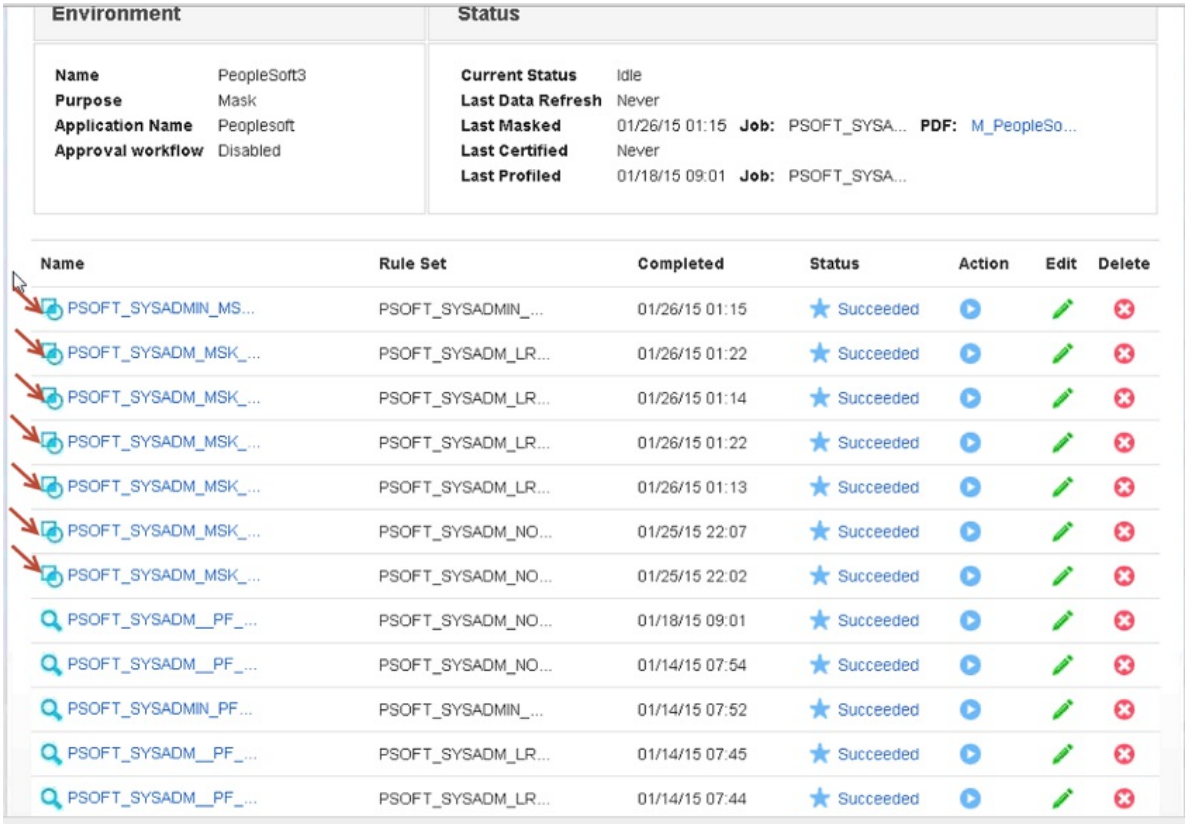
appears if a prescript file was already specified.) For information about creating your own prescript files, see Creating SQL Statements to Run Before and After Jobs.

xxii. Postscript — (optional) Specify the full pathname of a file that contains SQL statements to be run after the job finishes, or click Browse to specify a file. If you are editing the job and a postscript file is already specified, you can click the Delete button to remove the file. (The Delete button only appears if a postscript file was already specified.) For information about creating your own postscript files, see Creating SQL Statements to Run Before and After Jobs.

xxiii. Comments — (optional) Add comments related to this masking job.

xxiv. Email — (optional) Add e-mail address(es) to which to send status messages.

3. When you are finished, click Save.



Provision Masked VDBs

Masked virtual databases (VDBs) function just like normal VDBs. The only distinction is that the data they contain has been masked by a masking job. Masked VDBs can be replicated to a separate Delphix Engine (in non-prod) without sending the original data that was obfuscated during masking using a process called Selective Data Distribution (SDD). This topic describes how to work with masked VDBs.

Prerequisites

Before attempting to create a Masked VDB, you should be familiar with both Delphix Virtualization and Delphix Masking concepts and workflows.

Restrictions

- A single masking job cannot be assigned to multiple VDBs simultaneously. If you are using the same masking ruleset on multiple VDBs, be sure to create a unique job for each VDB to avoid any issues with provisioning or refreshing.
- Provisioning or refreshing masked VDBs is only supported for Oracle, MS SQL Server and Sysbase. Provisioning or refreshing other types of masked VDBs such as DB2 are not support.
- You cannot apply additional masking jobs to a masked VDB or its children.
- If a masking job has been applied to a VDB, you cannot create an unmasked snapshot of that VDB.
- Masking must take place during the process of provisioning a VDB. If an existing VDB has not had a masking job applied to it, then you cannot mask that particular VDB at any point in the future. All the data within the VDB and its parents will be accessible if it is replicated using SDD.
- When selecting a connector to use for Masked Provisioning, a "basic" connector must be used unless you are masking an Oracle Pluggable Database (PDB), in which case an "advanced" connector must be used.

Identifying and Navigating to Masked VDBs

Masked VDBs appear in the Virtualization Engine's Datasets pane, just like regular VDBs. They are most obviously identified by the different icon used to represent them. In addition, a masked VDBs Configuration tab will contain information about the masking job that you applied to it. Generally, anything you can do with an unmasked VDB is also possible with a

masked VDB.

Provisioning Masked VDBs

- In the Virtualization Engine, associate a masking job with a dSource.
- Use the dSource provision wizard to provision a VDB with a masking job.

Associating a Masking Job with the dSource

To provision a masked VDB, you must first indicate that the masking job you are using is complete and applicable to a particular database. You do this by associating the masking job with a dSource.

1. In the Datasets panel on the left-hand side of the screen, click the dSource to which the masking job is applicable and with which it will be associated.

2. Click the Configuration tab.

3. Click the Masking tab.

4. Click the pencil icon to edit. All masking jobs on this Delphix Engine that have not been associated with another dSource will be listed on the right-hand side.

5. Select the job you want to associate with this dSource.

6. Click the green checkmark to confirm.

7. Repeat for any other jobs that you want to associate with this dSource at this time.

The Delphix Engine now considers this masking job to be applicable to this dSource and ready for use. When provisioning from snapshots of this dSource, this masking job will now be available.

!!! note

```
Masking jobs can also be associated with virtual sources in addition to dSources.
```

Provisioning a Masked VDB using the dSource Provisioning Wizard

The steps required to provision a masked VDB are almost identical to the steps required to provision an unmasked VDB. Once you have created a masked VDB, you cannot un-mask it, nor can you alter which masking job it uses. All snapshots in the VDBs TimeFlow will always be masked using the masking method that you selected when you provisioned the masked VDB.

1. In the Datasets panel on the left-hand side of the screen, select the dSource.
2. Click the TimeFlow tab.
3. Click Provision VDB icon.
4. Review the information for Installation Home, Database Unique Name, SID, and Database Name. Edit as necessary.
5. Review the Mount Base and Environment User. Edit as necessary.
    - If you want to use login credentials on the target environment that are different from the login credentials associated with the Environment User, select Specify Privileged Credentials.
6. Click Next.
7. If necessary, edit the Target Group for the VDB.
8. Select the None option for the Snapshot Policy for theVDB .

!!! note "Snapshot Policy Selection"

```
For almost all use cases involving Masked VDBs, a Snapshot Policy of None is appropriate. U
```

9. Click Next.
10. Click Mask this VDB. You will be presented with two options to mask this VDB:
    - Select an existing masking job: Choose this option if you want to mask using preconfigured Masking Job. Only masking jobs that have been associated with the parent dSource will be available.

!!! note "Selecting Unique Masking Jobs"

```
If you are using the same masking ruleset on multiple VDBs, be sure to create a unique job
```

- Masking using scripts(s): Alternatively, you may define some Configure Clone scripts in the Hooks step to perform masking.

!!! note "Defining Configure Clone Hooks to Mask VDB"

```
If you choose to mask using script(s), you must define the Configure Clone hooks to run mas
```

11. Click Next.

12. Specify any Pre or Post Scripts that should be used during the provisioning process. If the VDB was configured before running the masking job using scripts that impact either user access or the database schema, those same scripts should also be used here. Be sure to define the Configure Clone hooks to run the masking job if you choose to mask using script(s) in the Masking step.
    

13. Click Next.

14. Click Submit.

If you click Actions in the the upper right-hand corner, the Actions sidebar will appear and list an action indicating that masking is running. You can verify this and monitor progress by going to the Masking Engine page and clicking the Monitor tab.

!!! note

```
Once you have created a masked VDB, you can provision its masked data to create additional
```

Refresh a Masked VDB

You refresh a masked VDB in exactly the same way as you refresh a normal VDB. As with provisioning a masked VDB, the masking job will be run during the refresh process.

1. Login to the Delphix Management application.
2. Click Manage.
3. Select Datasets.
4. Select the VDB you want to refresh.
5. Click the Refresh VDB button (2 circular arrows).
6. Select More Accurate and Next.
7. Select desired refresh point snapshot or click the eye icon to choose Latest available range, A point in time, or An SCN to refresh from.
8. Click Next.
9. Click Submit to confirm.
10. Click the Actions link to watch the progress of the refresh job.
11. To see when the VDB was last refreshed/provisioned, check the Time Point on the Status page.

Disassociating a Masking Operation on a dSource

If a masking job is found to be unsuitable or should be retired, you can disassociate it though the same database card that you used to associate it.

1. Deselect the job.
2. Click green arrow to confirm.
   Note that this will only prevent the creation of new masked VDBs with this job. It will not alter existing masked VDBs in any way. When disassociating a job, review the existing masked VDBs and consider whether you need to delete or disable any of them.

Masked VDB Data Operations

The following data operations are available to masked VDBs:

- Rewind : Alter the database to contain masked data from a previous point in time.
- Refresh : Get new data from the parent dSouce and mask it.
- Disable : Turn off the database and remove it from the host system.
- Enable : Turn on the database and make it available on the host system.

Virtualization and Masking Engine Compatibility Matrix

	Masking Engine Version

Virtualization Engine Version	
5.0 releases	5.0 releases (minor versions do not need to match)
5.1 releases	5.1 releases (minor versions do not need to match)
5.2 releases	5.2 releases (minor versions do not need to match)
5.2.5.0 (or later 5.2 minor release)	5.2.5.0 (or later 5.2 minor release)
5.3 releases	5.3 releases (minor versions do not need to match)

Monitoring Masking Job

This section describes how users can monitor the progress of a masking job.

Running and Stopping Jobs from the Environment Overview Screen

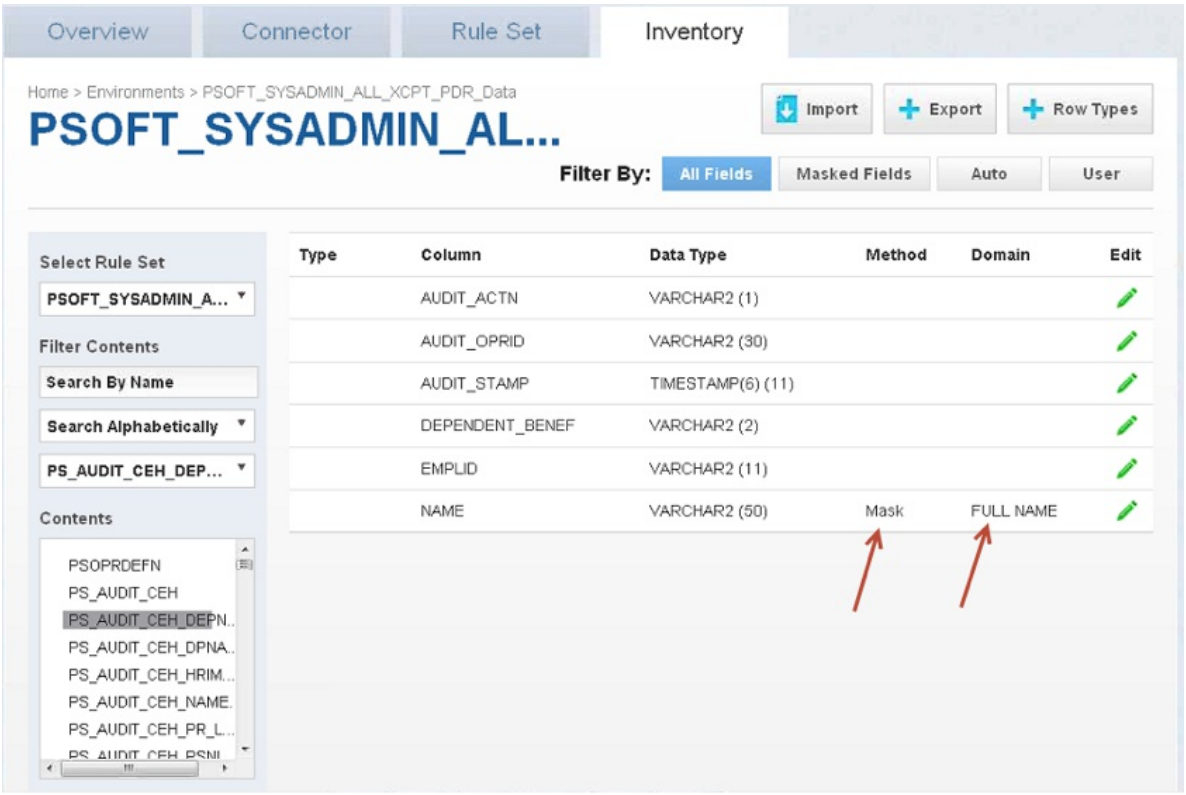
To run or rerun a job from the Environment Overview screen:

- Click the Run icon (play icon) in the Action column for the desired job.

The Run icon changes to a Stop icon while the job is running. When the job is complete, the Status changes.

To stop a running job from the Environment Overview screen:

1. Locate the job you want to stop.
2. In the job's Action column, click the Stop icon.
3. A popup appears asking, "Are you sure you want to stop job?" Click OK.
4. When the job has been stopped, its status changes.
5. After the job completes successfully, return to the Inventory and check that the Domain and Method populated automatically for sensitive data. Sample screenshot below.


Overview
Connector
Rule Set
Inventory
Home > Environments > PSOFT_SYSADMIN_ALL_XCPT_PDR_Data
PSOFT_SYSADMIN_AL...
Import
Export
Row Types
Filter By:
All Fields
Masked Fields
Auto
User
Select Rule Set
PSOFT_SYSADMIN_A...
Filter Contents
Search By Name
Search Alphabetically
PS_AUDIT_CEH_DEP...
Contents
PSOPRDEFN
PS_AUDIT_CEH
PS_AUDIT_CEH_DEPN..
PS_AUDIT_CEH_DPNA..
PS_AUDIT_CEH_HRIM...
PS_AUDIT_CEH_NAME.
PS_AUDIT_CEH_PR_L...
Type
Column
Data Type
Method
Domain
Edit
AUDIT_ACTN
VARCHAR2 (1)
AUDIT_OPRID
VARCHAR2 (30)
AUDIT_STAMP
TIMESTAMP(6) (11)
DEPENDENT_BENEF
VARCHAR2 (2)
EMPLID
VARCHAR2 (11)
NAME
VARCHAR2 (50)
Mask
FULL NAME

Masking Job Wizard

The Delphix Masking job wizard enables users to create and modify masking jobs. While the wizard facilitates a number of workflows and operations, more advanced functionality and a finer control of features is available directly in the masking application. The Job Wizard currently functions only with certain data platforms, but these constraints do not apply when working directly in the masking application.

Supported Data Platforms

The following data platforms are currently supported from within the Job Wizard:

- Oracle Database
- RDS Oracle Database
- MSSQL Server Database
- Sybase Database

This restricted list only affects your use of the wizard; an expanded number of platforms are supported directly in the masking application. Some operations within the Job Wizard are also limited. See below for details.

Supported Operations

While creating a masking job in the Job Wizard, you are able to do the following:

- Create a new application or use an existing application
- Create a new environment or use an existing environment
- Create a new connector*
- Create a new rule set
- Update inventory*
- Create a masking job*
- Update a masking job
- Change the connector for an existing job
- Change the rule set for an existing connector
- Run a newly created job immediately
- Run an updated job immediately after the update

!!! note

Operations marked with an asterisk are limited in the Job Wizard but fully supported in the main application.

What is Not Supported in the Wizard

The following data platforms and operations are not supported in the Job Wizard.
To access additional functionality, use the main masking application.

Unsupported Data Types

The following data types are supported when using the main masking application but are not currently supported in the Job Wizard:

- DB2 Database
- PostgreSQL Database
- Generic Database
- Delimited File
- Excel Sheet File
- Fixed File
- Mainframe File
- XML File

Unsupported Operations

The following operations are not yet supported from within the Job Wizard:

- Creating any connector or rule set for an unsupported data type
- Deleting any application, environment, connector, rule set, or masking job
- Importing or exporting any object
- Updating an environment
- Creating a connector using Advanced mode
- Updating a connector
- Updating a rule set
- Creating a job for an unsupported data type
- Modifying a job for an unsupported data type
- Monitoring running jobs
- Creating, editing, deleting, or running any Profile jobs

Opening the Masking Job Wizard

When you first login to masking, the welcome screen offers a link to learn more or begin masking immediately. To open the Job Wizard, click Run on the welcome page.

To use the Job Wizard from the masking application, click the Create Job button in the upper right-hand corner, as highlighted in the screenshot below.

Creating a New Masking Job

The Job Wizard makes creating a new masking job much easier by guiding you through the process. You can create new objects or choose to use existing ones that have already been defined. When creating a new masking job, the Job Wizard follows this sequence:

- Job Naming
- Application/Environment Selection
- Connection Selection
- Rule Set Selection
- Inventory Selection
- Summary Page

You can navigate back and forth through the pages of the Job Wizard.

!!! note
If the product times out due to long inactivity, you will need to start over. |

To create a new masking Job using the new Job Wizard, follow the procedure below:

1. Log into your Delphix Masking Engine and from the Welcome screen select Run.
2. Select the New radio button and enter a name for your Masking job.
   
3. Click Next.
4. From the drop-down menu select an Application and Environment. If none exist use the Add button to add one.
   
5. Click Next.
6. Select a Connector from the drop-down menu. If none exists select the Add button, then

use the Add Connector dialog to add a new connector. The Job Wizard only supports the following Connector types:

- Database - MS SQL
- Database - Oracle
- Database - RDS Oracle
- Database - Sybase

7.Click Next.

7. On the Rule Set screen select an existing Rule set or create a new one by clicking the Add button.

8. Click Next.
9. From the Inventory screen select how your data will be masked. In the screenshot below we are masking subscriber last names.

10. Click Next.
11. The final screen of the Job Wizard displays a Summary of your selections.

12. Clicking Run Masking Job Now and go to Monitor progress, saves your job and runs it immediately. Save Job allows you to save your job and run it at a later date. Note: Selecting this option means your data will not be masked until you run the job.

When Objects Are Saved

Application, environment, connector, and rule set objects are created and persist after you click the Add button and see a success message. If you cancel the Job Wizard before completing the job setup, the objects you created will be saved, and they will be available for use the next time you launch the Job Wizard.

The Inventory definition is saved when you change the selection of a table or column, or when another View filter is applied.

The masking job is saved when you click either Save Job or Run Masking Job Now and go to Monitor progress and a success message is returned on the Summary screen.

Updating an Existing Masking Job

You can use the Job Wizard to modify any masking job that targets a supported

data type.

1. On the Job screen of the Job Wizard, select Modify Existing

2. From the list of available jobs slect the one you want to modify. This list only shows jobs that are supported in the wizard. You can filter the job list by selecting the filter icon .

3. Once you select a job, you can change the following as part of the Modify flow:

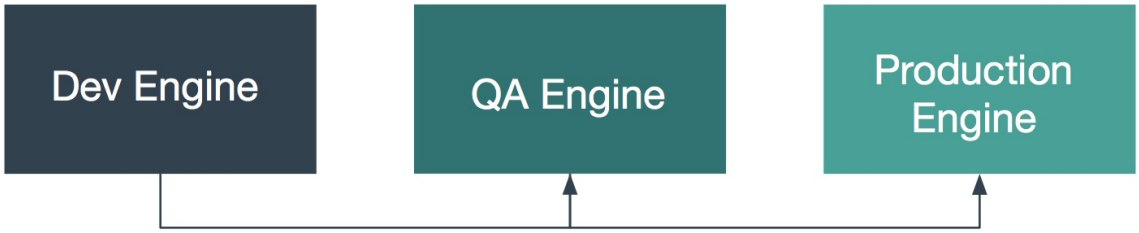
    - Change/create new connector
    - Change/create new rule set
    - Update inventory
    - Save or run the modified job

    You cannot alter application and environment settings as part of the Modify flow, but you can do so in the main masking application.

Your organization may have more than one masking engine, and in certain circumstances, it may want to coordinate the operation of those engines. In particular, there are two specific scenarios in which an organization could benefit from some level of interaction and orchestration between multiple masking engines.

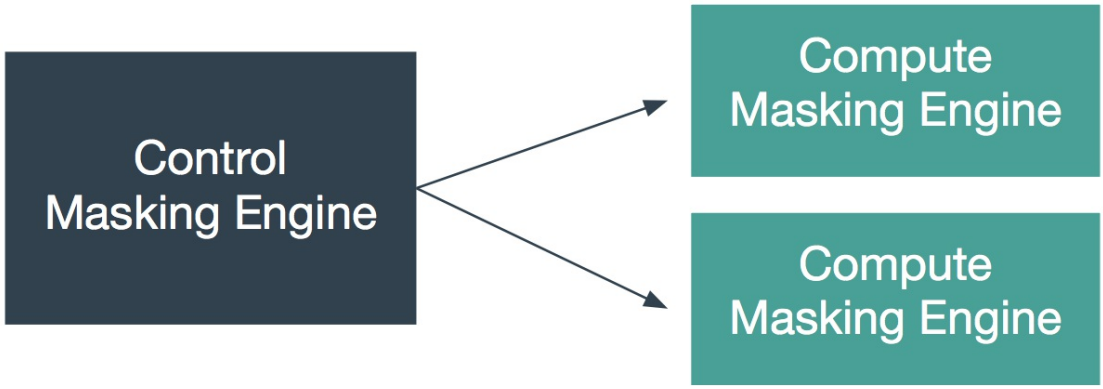
Software Development Life Cycle (SDLC)

Using an SDLC process often requires setting up multiple masking engines, each for a different part of the cycle (Development, QA, Production).



Distributed Execution

For many organizations, the size of the profiling and masking workloads requires more than one production masking engine. These masking engines can be identical in configuration or be partially equivalent depending on the organization's needs.



For both of these use cases, you will need to be able to move various objects between masking engines. These objects may include the following:

- Algorithms
- Connectors
- Domains
- File Formats
- Inventories
- Masking Jobs
- Profile Expressions
- Profile Jobs
- Profile Sets
- Rulesets

You can move a subset of these objects between engines using the Masking V5 APIs. See the following sections for instructions.

Best Practice Guide & Example Architectures for Synchronizing

Engine synchronization provides a general and flexible way to move masking algorithms and objects necessary to run an identical job on another engine. It is recommended that the syncable objects move in only one direction. That is, objects should be exported from one engine and imported into others but should not go in the other direction. This recommendation is primarily to simplify management of which objects exist on which engine.

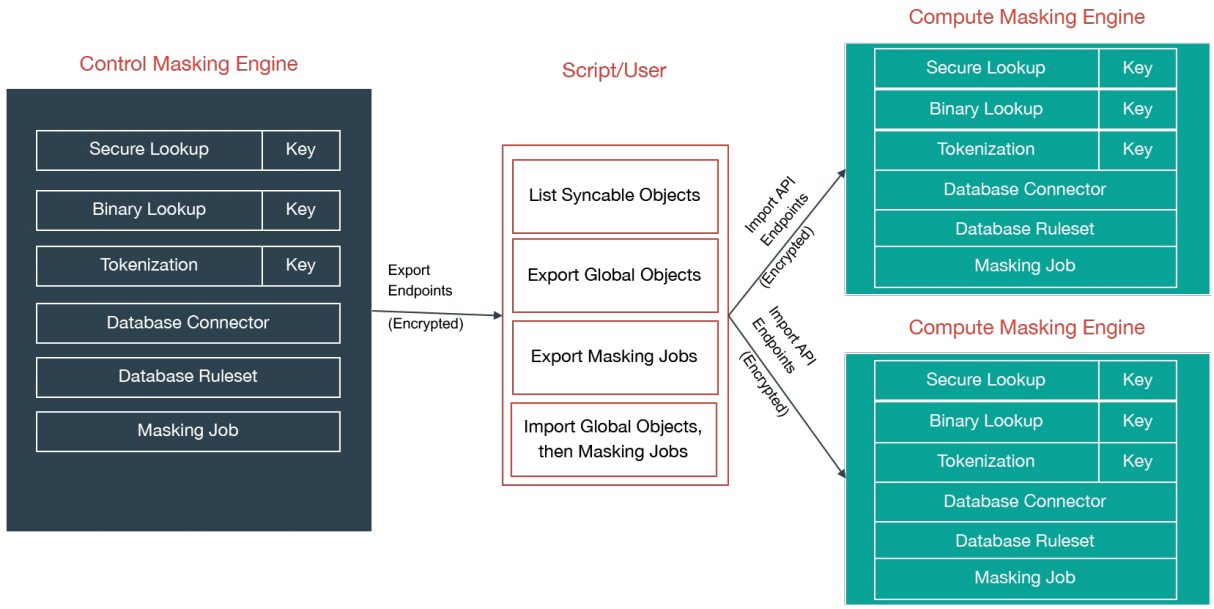
Two example architectures are described below. Note that the two architectures could be combined by having multiple production engines instead of a single one.

Horizontal Scale

The first architecture aims to address the problem of horizontal scale -- that is, achieving consistent masking across a large data estate by deploying multiple masking engines. In this architecture, syncable objects are authored on one engine, labeled “Control Masking Engine” in the diagram below. Those objects are then distributed to “Compute Masking Engines” using the engine synchronization APIs. The synchronized

algorithms and masking jobs will produce the same masked output on all of the engines, thus enabling large data estates to be masked consistently.

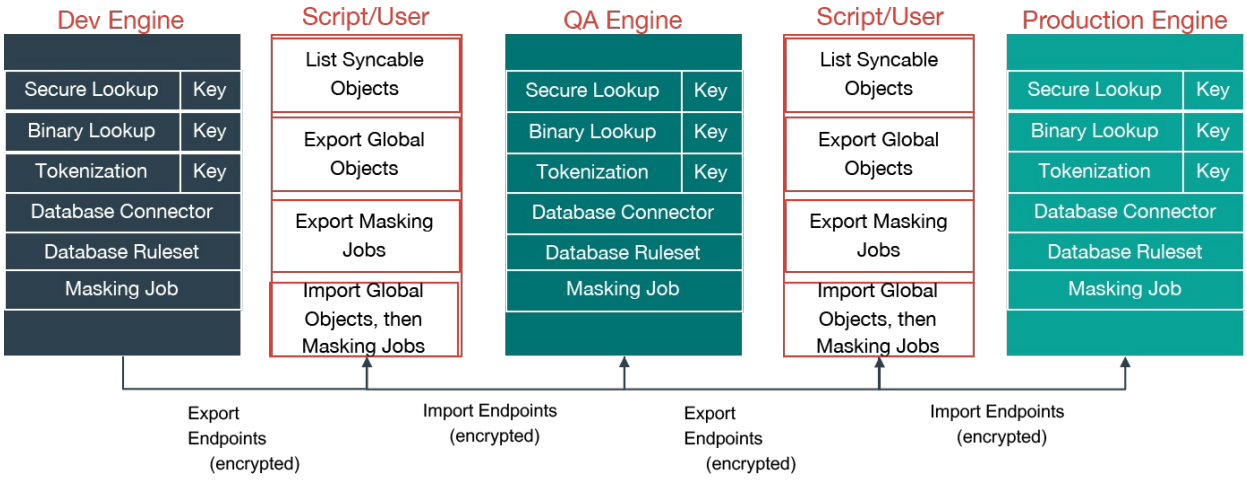
Horizontal Scale Use Case



SDLC

The second architecture addresses the desire to author algorithms on one engine, to test and certify them on another, and finally to deploy them to a production engine. Here, algorithms are authored on the first engine, labeled “Dev Engine” in the diagram below. When the developer is satisfied, the algorithms are exported from the Dev Engine and imported to the QA Engine where they can be tested and certified. Finally, they are exported from the QA engine and imported to the production engine.

SDLC (Algorithm) Use Case



Masking API Call Concepts

Syncable object

Syncable objects are external representations of objects within the masking engine that can be exported from one engine and imported into another. EngineSync currently supports exporting a subset of algorithms, the encryption key and all the objects necessary for a masking job.
Note: We do not currently support Mainframe masking jobs.

Object Identifiers and Types

EngineSync uses object identifiers to name unique objects within the engine. The follow object types are currently supported:

- DATABASE_CONNECTOR
- DATABASE_RULESET
- DOMAIN
- FILE_CONNECTOR
- FILE_FORMAT
- FILE_RULESET
- GLOBAL_OBJECT
- KEY
- Certain algorithms:
  - BINARYLOOKUP
  - CLEANSING
  - DATE_SHIFT
  - LOOKUP
  - MIN_MAX
  - REDACTION
  - SEGMENT
  - TOKENIZATION
  - MAPPLET
- MASKING_JOB
- PROFILE_EXPRESSION
- PROFILE_JOB
- PROFILE_SET

The following lists the object types that are simply for the purpose of

referencing a particular state of the exported object. These are not meant to be exported by request. The functions of these are further explained in the latter sections.

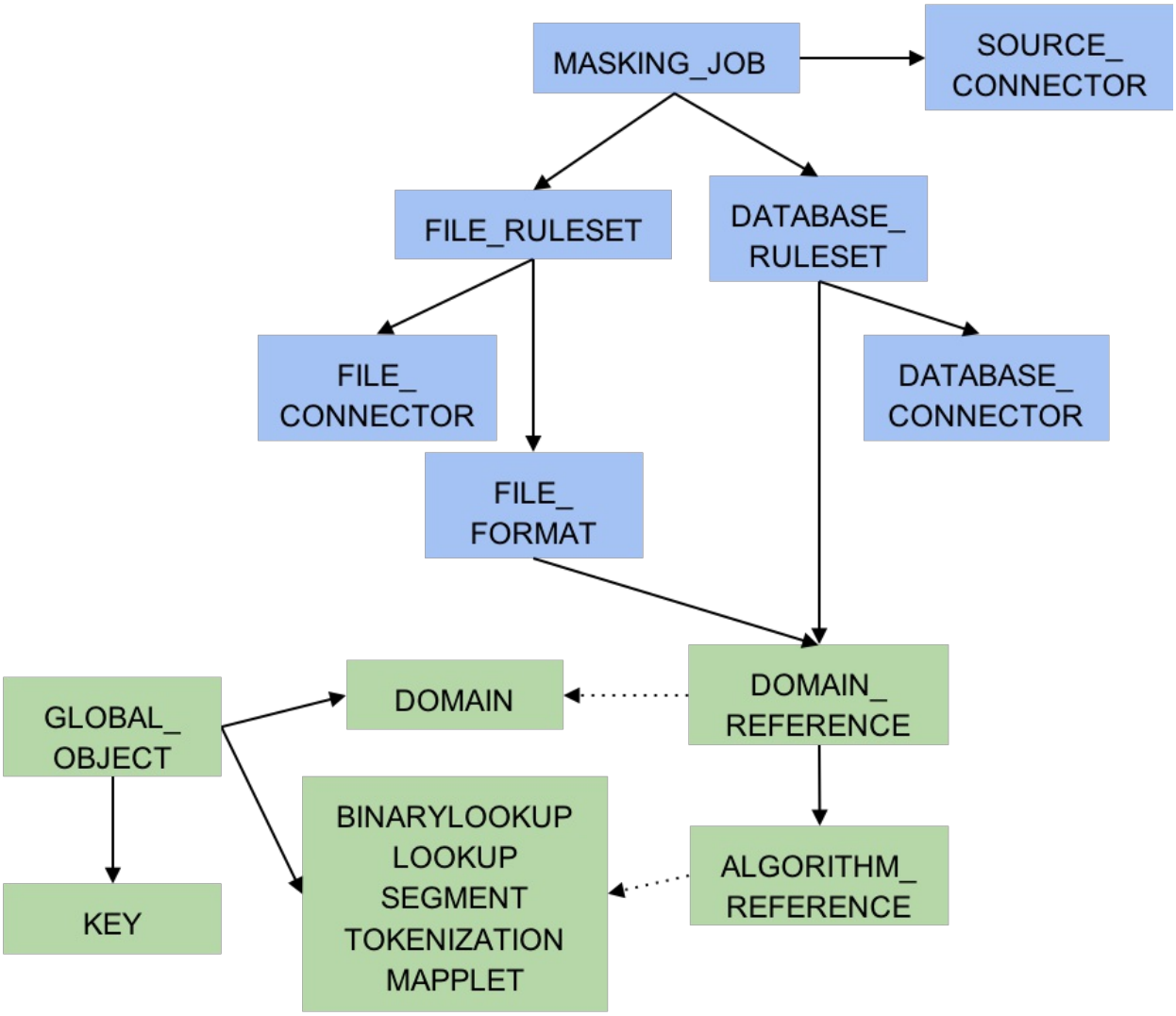
- ALGORITHM_REFERENCE
- DOMAIN_REFERENCE
- PROFILE_EXPRESSION_REFERENCE
- PROFILE_SET_REFERENCE
- SOURCE_DATABASE_CONNECTOR
- SOURCE_FILE_CONNECTOR

Dependencies

Most objects within the Masking Engine are compositional. In order to properly capture the behavior of a syncable object, you must export its dependencies along with the object itself. Fortunately, all the necessary dependencies are exported along with the object you request; thus, it is not something you need to keep track of and worry about.

Syncable Object dependencies relationship

Note: Green represents global objects (objects that are central to the entire engine), and blue represents objects that need to be a part of an environment



Object Revision Tracking

The revision hash is used to help you determine whether the behavior of a syncable object is the
same between engines. Because objects within the Masking Engine are compositional, the behavior of an object is influenced by all of its dependencies. When a syncable object is listed or exported, the Masking Engine computes a revision_hash, which uniquely identifies the object's behavior.

The revision_hash is a SHA1 hash that represents that object's state, as well as the state of all objects it depends on. If two objects have the same revision hash, it is safe to assume that the behavior the objects is the same. However, it is possible for two objects to have the same behavior but have divergent revision hashes. For example, you could have two lookup algorithms with the same name, lookup file, and key, and they do not necessarily guaranteed to have the same revision hash.

!!! note
The revision_hash does not change when the password or the ssh key for either the FILE_CONNECTOR or DATABASE_CONNECTOR is updated. This is intentionally done because we do not export the password or the ssh key for security purposes. This allows users to update the password after import without changing the revision_hash. If a user is overriding a connector that already has a password set, the import does not reset the password and will leave the current, pre-import value.

Export Document

You can export one or more syncable objects that are listed in the */syncable-objects* endpoint. The export document will include the set of objects that you requested for export and all of their dependencies that are required to properly import those objects into another engine.

The export document is exported as an opaque blob. Do not edit it outside of the Masking Engine.

Export Document Encryption

You can request that the export document be encrypted using a passphrase. Once the document is encrypted with the passphrase, the engine forgets the passphrase. You will need to provide the same passphrase during import to decrypt the document.

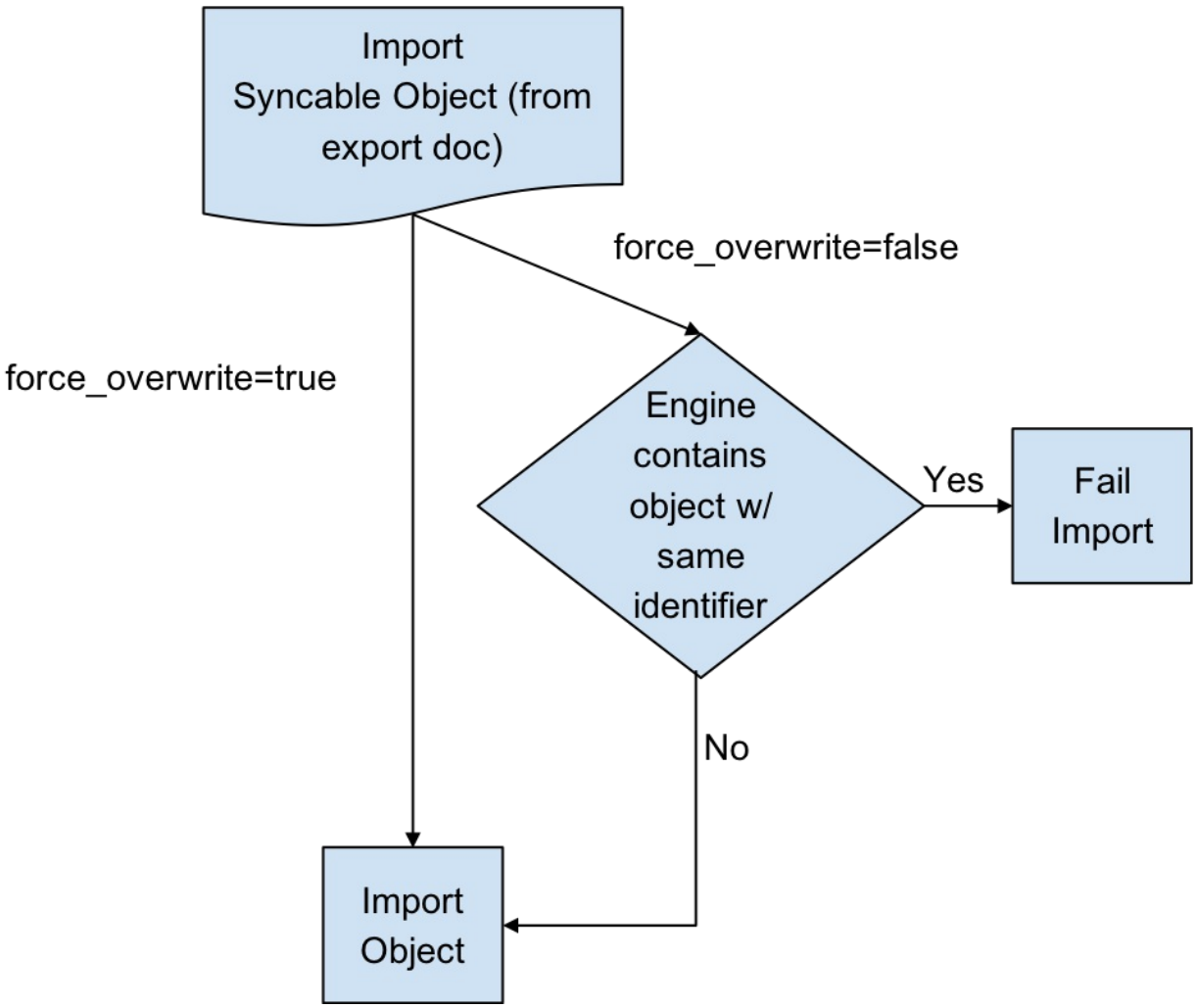
Digital Signature

In order to detect accidental or malicious modification of the export document, each document is digitally signed. If the export document does not match its expected digital signature, a Masking Engine will not import the document.

Overwrite

When an object to be imported has the same name as a currently existing object, importing it will cause the other object to be changed. Since this might not be intended, we offer a flag called force_overwrite. If force_overwrite is set to false and doing the import will change an existing object on the masking engine, we fail the import. This workflow

is shown below.



Attempting to Import Identical Objects

The Masking Engine checks for the existence of the same object contents during the import of an object. If it is determined that the engine and the document being imported contain the same content, a result of SUCCESS will be returned without repeating the work of a full import. For example, importing an entire ruleset with hundreds of thousands of tables can be quite time consuming, and this should not be repeated if the same object already exists. If the object content matches and we skip the full import we note this in the application log.

Below is an example log statement when an identical database connector was imported:

```
2017-07-19 10:17:06,075 [http-nio-8282-exec-4] INFO
c.d.s.marshallers.SyncableMarshaller - Skipping import process for
{
```

```
"objectType": "DATABASE_CONNECTOR",
"id": {
"@type": "type.googleapis.com/IntegerIdentifier",
"id": 1
}
}, due to no discrepancy between the existing and importing object
```

####

Depending on the object type, some define an object by a String (name) and some by an Integer (object id). Objects that can have the same name in multiple environments, such as connectors, rulesets, and masking jobs, are exported based on a unique id associated with them. Global objects, which do not have overlapping names, are exported and identified based on their names. Something to note here is that objects exported based on their ids will overwrite the object with the *same name* rather than the same id. This means that for all importing objects, we define the identity of an object to be based on the name in the same environment.

For example, if I export a database connector named *testConnector* with the following export object metadata:

```
{
"objectIdentifier": {
"id": 5
},
"objectType": "DATABASE_CONNECTOR",
"revisionHash": "68eaffef400e426520a5fcbb683419db3be53317"
}
```

And then I import this object into some engine's environment with the following list of connectors:

id	connector name	more information
1	testConnector	…
5	otherConnector	…

testConnector of id 1 will be overwritten, instead of *otherConnector*.

Overwrite of the Encryption Key

The global encryption key is somewhat special in that it always exists. Specifying *force_overwrite=false* will always fail to import the encryption key unless the encryption key has been previously synchronized using *force_overwrite=true*.

Specifying *force_overwrite=true* will always overwrite the engine's encryption key with the contents of the encryption key in the export document.

Error handling

Export documents often have multiple objects to be imported at once. For example, when exporting a database ruleset, you will export both the database ruleset and the database connector since a ruleset depends on a connector.

The engine will import one object at a time, where the dependencies are imported first. If there is an error importing an object, the import process will abort and all objects that have successfully been imported during this request will get rolled back. For example, say you are importing objects A, B, and C. Import successfully imports A. During the import of B, the engine encounters an error. The import of A will roll back, and import of C will never execute. This will leave the engine in a state identical to the one it was in prior to the failed import.

Concurrent Sync Operations

To prevent race conditions with concurrent imports and jobs running, we currently do not allow concurrent import operations. We also do not allow imports while masking jobs or exports are running. It is best to do imports when a machine is not running jobs or other exports in order to guarantee that the final state of each of those operations is as expected. If they are done at the same time, the operations will fail with relevant error messages.

Global Objects

GLOBAL_OBJECT is a syncable object type that is a collection of all syncable algorithms, DOMAIN(s), PROFILE_SET(s), PROFILE_EXPRESSION(s) and KEY (global key). This represents objects in the Masking Engine that are available across all environments, and are not a part of any specific environment. When a user requests to export GLOBAL_OBJECT, every syncable algorithm, profile set, profile expression and domain on the engine will be exported as the bundle. If a DOMAIN, PROFILE_SET, or PROFILE_EXPRESSION has a dependency on a non-syncable algorithm, such as Mapping, it will not be exported.

This separation was added because global objects 1) containing large lookup files are projected to be time consuming and 2) are expected to be synchronized much less frequently than any masking job related metadata. Examples on how to use it will be available in the Example User Workflow section.

References Objects

As mentioned in the *Global Objects* section, we expect the users to synchronize global objects and masking jobs at different frequencies. To avoid any unnecessary export of large algorithms, any objects (MASKING_JOB, PROFILE_JOB, DATABASE_RULESET, FILE_FORMAT and FILE_RULESET) that have dependencies on algorithms will export just the references to the objects by default. This way we check whether the necessary dependency exists on the importing engine by comparing the references; if not, we fail the import execution with an appropriate message. Domains, profile sets, and profile expressions are the exception to this. Exporting any of these objects will also export the full algorithm.

On-The-Fly Masking Jobs

By definition, On-The-Fly (OTF) masking jobs work with a source environment/connector and a target environment/connector, masking the data from the source connector into that of the target connector. With masking jobs, a target *environment_id* is always required to specify which environment to import the job and its target connector. In addition to the target *environment_id,* OTF masking jobs require the

specification of a *source_environment_id* into which to import the source connector. The source connector is copied into the specified source environment (*source_environment_id*), and is represented by the SOURCE_DATABASE_CONNECTOR or

SOURCE_FILE_CONNECTOR for database and file masking jobs respectively in the export document. These source connectors are virtually identical to their DATABASE_CONNECTOR and FILE_CONNECTOR counterparts, but are represented differently in the OTF jobs to distinguish them from the target connector (i.e., DATABASE_CONNECTOR or FILE_CONNECTOR).

Circular Dependencies

It is possible to have a set of objects that end up depending on each other. This would be the case if a PROFILE_SET depended on a PROFILE_EXPRESSION that depended on a DOMAIN that depended on a REDACTION algorithm that depended on the original PROFILE_SET. The masking application will detect such scenarios on export and refuse to export such configurations.

This can be worked around by creating a second PROFILE_SET that contains PROFILE_EXPRESSIONS that do not depend on a DOMAIN that depends on a REDACTION algorithm. Simply ensure that the regular expressions are the same in the newly created PROFILE_EXPRESSIONs and assign the REDACTION algorithm to the new PROFILE_SET instead. The REDACTION algorithm will function the same but the dependency loop will have been broken.

GET /syncable-objects[?object_type=<type\>]

This endpoint lists all objects in an engine that are syncable and can be exported. Any object which can be exported, can be imported into another engine. The endpoint takes an optional parameter to filter by a specific object type. Each object is listed with its revision_hash.
Note that if a syncable object depends on a non-syncable object (i.e. DOMAIN using a mapping algorithm), it will say so in the “revisionHash” attribute, and will not be exportable.

Example CURL command:

```
curl -X GET
--header 'Accept: application/json'
--header 'Authorization: 21c45f0e-82f4-4b04-9072-b49072986231'
'http://masking-engine.com:8282/masking/api/syncable-objects?page_number=1'
```

POST /export

This endpoint allows you to export one or more objects in batch fashion. The result of the export is an export document and a set of metadata that describes what was exported. You are expected to specify which objects to export by copying their object identifiers from the /syncable-objects endpoint.

The endpoint has a single optional header, *passphrase*. If you provide the passphrase, the export document will be encrypted using it.

Example CURL command:

```
curl -X POST
--header 'Content-Type: application/json'
--header 'Accept: application/json'
--header 'Authorization: 21c45f0e-82f4-4b04-9072-b49072986231'
-d '[
{
"objectIdentifier": {“id”: 1},
"objectType": "MASKING_JOB",
"revisionHash": "asdfjkl12jijfdsaklfj21ojasdk"
}
]'
'http://masking-engine.com:8282/masking/api/export'
```

POST /export-async

This endpoint does exactly the same thing as /export, but the execution is done asynchronously. The response returns an async task in the form of this:

```
{
"asyncTaskId": 66,
"operation": "EXPORT",
"reference": "EXPORT-ZXhwb3J0X2RvY3VtZW50XzJjcm1EV09yLmpzb24=",
"status": "RUNNING",
"startTime": "2018-04-13T17:49:55.354+0000",
"cancellable": false
}
```

Example CURL command:

```
curl -s -X POST
--header 'Content-Type: application/json'
--header 'Accept: application/json'
--header 'Authorization: 21c45f0e-82f4-4b04-9072-b49072986231'
-d "[
{
"objectIdentifier": {“id”: 1},
"objectType": "MASKING_JOB",
"revisionHash": "asdfjkl12jijfdsaklfj21ojasdk"
}
]"
"http://masking-engine.com:8282/masking/api/export-async"
```

The *reference* is used to retrieve the export document of completed async export tasks from the /file-downloads endpoint. The downloaded file from this reference should look exactly the same as the response from /export.

Example CURL command:

```
curl -s -X GET
--header 'Accept: application/octet-stream'
--header 'Authorization: 21c45f0e-82f4-4b04-9072-b49072986231'
-o "<OUTPUT_FILE_PATH>" "http://masking-engine.com:8282/masking/api/file-downloads/EXPORT-Z
```

Error handling

If an error occurs while exporting one or more elements in the export document, the entire export will
abort.

POST /import

```
POST /import?force_overwrite=<true|false>[&environment_id=<id>][&source_environment_id=<id>
```

This endpoint allows you to import a document exported from another engine. The response returns a list of objects that were imported and whether the import was successful.

The endpoint has one required parameter, *force_overwrite*, two optional parameters *environment_id* and *source_environment_id*, and an optional HTTP header, *passphrase*, which if provided, will cause the engine to attempt to decrypt the document using the specified passphrase. The required *force_overwrite* parameter dictates how to deal with conflicting objects. *environment_id* is necessary for all non-global objects that need to belong in an environment. *source_environment_id* is used for On-The-Fly masking jobs.

Example CURL command:

```
curl -X POST
--header 'Content-Type: application/json'
--header 'Accept: application/json'
--header 'Authorization: 21c45f0e-82f4-4b04-9072-b49072986231'
-d '{
"exportResponseMetadata": {
"exportHost": "masking-engine.com:8282",
"exportDate": "Mon Aug 13 16:29:30 UTC 2018",
"exportedObjectList": [
{
"objectIdentifier": {
"algorithmName": "lookup_alg"
},
```

```
"objectType": "LOOKUP",
"revisionHash": "cf84d82c21f0e9d4105d37ae7979c0848486d861"
},
{
"objectIdentifier": {
"keyId": "global"
},
"objectType": "KEY",
"revisionHash": "1d8e9bc552d3ca1dcd218f9e197ea3955ccc29be"
}
]
},
"blob": "<OMITTED>",
"signature": "<OMITTED>", \
"publicKey": "<OMITTED>" \
}'
'http://masking-engine.com:8282/masking/api/import?force_overwrite=true'
```

POST /import-async

```
POST /import-async?force_overwrite=<true|false>[&environment_id=<id>][&source_environment_i
```

This endpoint does exactly the same thing as /import, but the execution is done asynchronously and the body is taken in as a file. The response returns an async task in the form of this:

```
{
"asyncTaskId": 67,
"operation": "IMPORT",
"reference": "IMPORT-ZXhwb3J0X2RvY3VtZW50XzJjcm1EV09yLmpzb24=",
"status": "RUNNING",
"startTime": "2018-04-13T17:49:55.354+0000",
"cancellable": false
}
```

The *reference* is used to retrieve the import status of completed async import tasks from the /file-downloads endpoint. The downloaded file from this reference should look exactly the same as the response from /import.

Example CURL command:

```
curl -s -X POST
--header 'Content-Type: multipart/form-data'
--header 'Accept: application/json'
--header 'Authorization: 21c45f0e-82f4-4b04-9072-b49072986231'
-F "file=@<DOWNLOADED_FILE_PATH>"
"http://masking-engine.com:8282/masking/api/import-async?force_overwrite=true"
```

One important piece of data used by many masking algorithms is the key, which determines the masked outcome of some value. Changing the key changes the output of these algorithms. For example, if the FIRST NAME algorithm masks “Michelle” to “Rachael,” changing the key might cause it to mask “Michelle” to “Ben”. There are two types of keys that the algorithms can depend on: either 1) global key or 2) individual key.

Global key

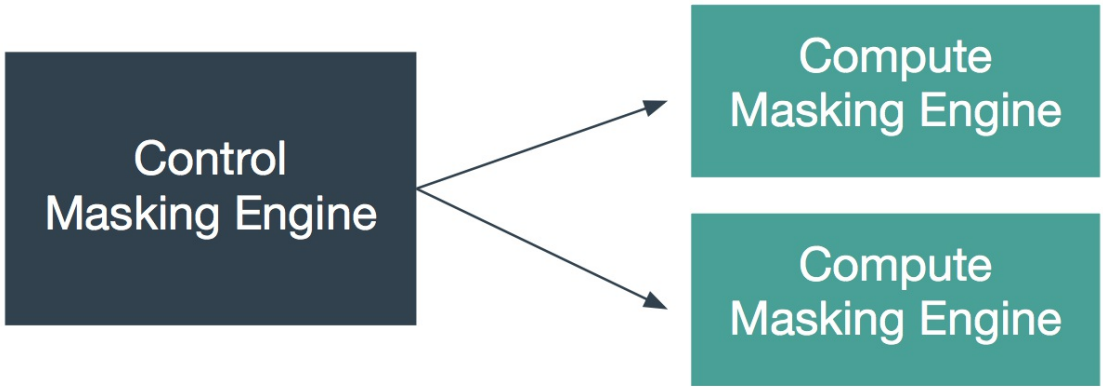
The following algorithm types depend on the global key for consistent masked results:

Custom Algorithm* (MAPPLET)

!!! note
A Custom Algorithm does not depend on the global key by nature. However, most mapplets currently used are implemented to use the global key.

A user with Administrator privileges can change the key by clicking the Generate New Key button in the Admin
tab.

!!! tip
Other actions are not allowed during the key generation process. Wait for the Generate New Key process to complete and a success dialogue to display in the user interface before performing additional actions on the Masking Engine (e.g., running a masking job).



Synchronizing the Global Key between Multiple Engines

In order for Custom Algorithms to behave the same way across several engines, all of those engines must have the same global key. Changing an engine's global key alters the behavior of all of the algorithms that depend on the global key.

You may want to change the key from time to time as a security management practice. If so, change it on all of the engines at the same time. That is, generate a new key on one engine, export that key, and import it to all of the other engines in the deployment.

Keys can be imported and exported independently of algorithms. To export the key from an engine, login to the engine through the login endpoint and then call export with the body shown below. Like all objects, you can encrypt the payload by supplying a passphrase header.

```
[{
"objectIdentifier": {
"keyId": "global"},
"objectType": "KEY"
}]
```

The API will return a JSON payload containing an encoded form of the key that you can install on other engines through the import endpoint. Like all exported objects, it is encoded in an opaque blob.

Individual Key

The following algorithm types have their own key that determines the masked results:

BINARYLOOKUP

DATE_SHIFT (only applies to DateShiftDiscrete)

LOOKUP

TOKENIZATION

The keys for each algorithm gets exported and imported with the algorithm itself, not separately. These individually associated keys can be randomized with an endpoint.

```
PUT http://masking-engine-A:8282/masking/api/algorithms/{algorithmName}/randomize-key
```

The following tables specify which algorithms are syncable between masking engines (in addition to the masking engine key).

!!! note
Only users with masking admin privilege are able to export and import algorithms.

User-defined Algorithms

Type	Syncable	Workaround
Lookup	Yes	NA
Binary Lookup	Yes	NA
Segmented Mapping	Yes	NA
Mapping	No	None
Tokenization	Yes	NA
Minmax	Yes	NA
Cleansing	Yes	NA
Free Text Redaction	Yes	NA
Custom Algorithm/Mapplet	Yes	NA (see “Custom Algorithm Syncability Guide” section)

Built-In Algorithms

Note that syncing built-in algorithms do not actually import the files associated with them but just updates their individual keys if they have them.

While some of the built in algorithms are not synchronizable, mainly due to them being non-deterministic, we still can support export of inventories that contain any built in algorithm. We just do not guarantee consistent masking of those non-synchronizable built in algorithms between engines.

Algorithm API Name	Algorithm UI Name	Type	Sy
AccNoLookup	ACCOUNT SL	lookup	Ye
AccountTK	ACCOUNT_TK	tokenization	Ye
AddrLine2Lookup	ADDRESS LINE 2 SL	lookup	Ye
AddrLookup	ADDRESS LINE SL	lookup	Ye
BusinessLegalEntityLookup	BUSINESS LEGAL ENTITY SL	lookup	Ye
CommentLookup	COMMENT SL	lookup	Ye
CreditCard	CREDIT CARD	calculated	No
DateShiftDiscrete	DATE SHIFT(DISCRETE)	calculated	Ye
DateShiftFixed	DATE SHIFT(FIXED)	calculated	No
DateShiftVariable	DATE SHIFT(VARIABLE)	calculated	No
DrivingLicenseNoLookup	DR LICENSE SL	lookup	Ye
DummyHospitalNameLookup	DUMMY_HOSPITAL_NAME_SL	lookup	Ye
EmailLookup	EMAIL SL	lookup	Ye
FirstNameLookup	FIRST NAME SL	lookup	Ye
FullNMLookup	FULL_NM_SL	lookup	Ye
LastNameLookup	LAST NAME SL	lookup	Ye
LastCommaFirstLookup	LAST_COMMA_FIRST_SL	lookup	Ye
NameTK	NAME_TK	tokenization	Ye
NullValueLookup	NULL SL	lookup	Ye
TelephoneNoLookup	PHONE SL	lookup	Ye
RandomValueLookup	RANDOM_VALUE_SL	lookup	Ye
SchoolNameLookup	SCHOOL NAME SL	lookup	Ye
SecureShuffle	SECURE SHUFFLE	calculated	No

SsnTK	SSN_TK	tokenization	Ye
USCountiesLookup	US_COUNTIES_SL	lookup	Ye
USCitiesLookup	USCITIES_SL	lookup	Ye
USstatecodesLookup	USSTATE_CODES_SL	lookup	Ye
USstatesLookup	USSTATES_SL	lookup	Ye
WebURLsLookup	WEB_URLS_SL	lookup	Ye
RepeatFirstDigit	ZIP+4	calculated	N

New Syncable Objects

We added the following new syncable objects in 5.3. Refer to the main documentation for more information on what they are, and how to use them.

- DATABASE_CONNECTOR
- DATABASE_RULESET
- DATE_SHIFT
- DOMAIN
- FILE_CONNECTOR
- FILE_FORMAT
- FILE_RULESET
- GLOBAL_OBJECT
- MASKING_JOB
- PROFILE_EXPRESSION (5.3.3.0)
- PROFILE_JOB (5.3.3.0)
- PROFILE_SET (5.3.3.0)

We also added the following new syncable algorithms in 5.3.

- CLEANSING (5.3.2.0)
- MIN_MAX (5.3.2.0)
- REDACTION (5.3.3.0)

Key per Algorithm

In pre-5.3, a global key for the engine was used by all algorithms that required a seed to determine the outcome of masked values. This included algorithms such as Lookup and Binary Lookup. Thus, in 5.2, exporting a Lookup Algorithm would automatically export the global encryption key as a dependency. In this release, we allow each algorithm to have its own independent key, exported as a part of the algorithm. Refer to the Key Management section for more detail.

Changed Model of Import Status Reporting

In 5.2, the import status looked like this:

```
{
"objectIdentifier": {
"keyId": "global"
},
"objectType": "KEY",
"importStatus": "SUCCESS"
}
```


Starting in Krypton, the import status of an object has extended to include the id or name it has imported into to reduce any confusion introduced with IntegerIdentifiers. For more information on the reason for this change, refer to Logic Behind Overwrite of IntegerIdentifier and StringIdentifier. For examples on what it now looks like, refer to the Example User Workflow section.

Changed Granularity of Transactions for Import

Starting in 5.3, an import of however many objects is performed as an atomic execution rather than using per-object atomicity. This means that the execution will either succeed at importing all objects or fail and import none at all. Refer to the Error Handling of Import logic flow diagram for more information.

Filter for /syncable-objects

Now that we have a large list of syncable objects, we have added a new feature for filtering based on the object type. Refer to the Endpoint page and the Example User Workflow section for more information.

Async Endpoints

Exporting a large MASKING_JOB with many dependencies can potentially take a long time. So we have decided to provide a new endpoint that exports and imports the objects asynchronously. Refer to the Endpoint section in the main documentation and the Example User Workflow page for more information.

This page provides some examples of some typical user workflows. More information on exactly how each endpoint works is available on the Endpoints page.

Syncing all Global Objects

The following steps can be used to sync all global objects from Masking Engine A to Masking Engine B. This will sync all algorithms and domains and should be done prior to syncing jobs or rulesets which might depend on them. For more information on the global object, see the Masking API Call Concepts section.

- On Masking Engine A, get the Authorization from the /login API

```
POST http://masking-engine-A:8282/masking/api/login

HEADER
Content-Type : application/json
Accept: application/json

BODY (raw)
{"username": "user123", "password": "pw123" }
```

Expected Result:

```
{ "Authorization": "dc2cff8b-e20d-4e28-8b7e-5d7c4aad0e2a" }
```

- On Masking Engine A, call GET /syncable-objects to get a list of syncable objects.

```
GET http://masking-engine-A:8282/masking/api/syncable-objects

HEADER
Authorization : dc2cff8b-e20d-4e28-8b7e-5d7c4aad0e2a (whatever you get from login)
Content-Type : application/json
Accept: application/json
```

Expected Result:

```
[
{
"objectIdentifier": {
```

```
"keyId": "global"
},
"objectType": "KEY",
"revisionHash": "68eaffef400e426520a5fcbb683419db3be53317"
},
{
"objectIdentifier": {
"id": 4
},
"objectType": "MASKING_JOB",
"revisionHash": "485343f1a68698497946f4f70d1cfdd76d516fd8"
},
{
"objectIdentifier": {
"algorithmName": "AddrLine2Lookup"
},
"objectType": "LOOKUP",
"revisionHash": "f397c46a97bddacf4203e35d7a538fda4bba6b12"
},
{
"objectIdentifier": {
"id": "global"
},
"objectType": "GLOBAL_OBJECT",
"revisionHash": "e230c46a97bddacf4201a35d7a538fda4bca6b14"
}
…
]
```


- On Masking EngineA, call /export-async on GLOBAL_OBJECT.

```
POST http://masking-engine-A:8282/masking/api/export-async

HEADER
Authorization : dc2cff8b-e20d-4e28-8b7e-5d7c4aad0e2a
Content-Type : application/json
Accept: application/json
passphrase (Optional): password to encrypt the export document

BODY
[
{
"objectIdentifier": {
"id": "global"
},
"objectType": "GLOBAL_OBJECT"
}
]
```

EXPECTED RESULT

```
{
"asyncTaskId": 2,
"operation": "EXPORT",
"reference": "EXPORT-ZXhwb3J0X2RvY3VtZW50Xzk0Wjlva3JDLmpzb24=",
"status": "RUNNING",
"startTime": "2018-06-15T20:36:35.483+0000",
"cancellable": false
}
```


- Download the export document with the reference above via the /file-download endpoint.

```
GET http://masking-engine-A:8282/masking/api/file-downloads/EXPORT-ZXhwb3J0X2RvY3VtZW50Xzk0

HEADER
Authorization : dc2cff8b-e20d-4e28-8b7e-5d7c4aad0e2a
Accept: application/octet-stream

EXPECTED RESULT
File - The exported document that would look identical to the response from /export with th
```

An example export document will look like this.


```
{
"exportResponseMetadata": {
"exportHost": "masking-engine-A:8282",
"exportDate": "Fri Jun 15 20:16:20 UTC 2018",
"requestedObjectList": [
{
"objectIdentifier": {
"id": "global"
},
"objectType": "GLOBAL_OBJECT",
"revisionHash": "579850b1c88baf74cee6bad61d81e2aa3dcc206c"
}
],
"exportedObjectList": [
{
"objectIdentifier": {
"id": "DRIVING_LC"
},
"objectType": "DOMAIN",
"revisionHash": "9ee90782488d14d369f9595dad7f593c961e785f"
},
{
"objectIdentifier": {
"algorithmName": "DrivingLicenseNoLookup"
},
```

```
"objectType": "LOOKUP",
"revisionHash": "e08ac9bfd4ed9f64d486cb47cdc07deb30ccc20f"
},
...
]
},
"blob":     "RAAAAAokZmZhNWIxNjktODMwMC00N2FlLWJjZmMtNjVhNDUzYWI3OTBjEhgyMDE4LTA2LTE1VDIwOj
"signature": "MCwCFAWGf/97wb+oYuSQizj8U12n7jpQAhQKGCaOJ4U8XyDAOEhMUWkzZXHrpw==",
"publicKey": "MIHxMIGoBgcqhkjOOAQBMIGcAkEA/KaCzo4Syrom78z3EQ5SbbB4sF7ey80etKII864WF64B81uRp
}
```


- On Masking Engine B, use the import-async endpoint to import the document downloaded from engine A.

```
POST http://masking-engine-B:8282/masking/api/import-async?force_overwrite=true
HEADER
Authorization : dc2cff8b-e20d-4e28-8b7e-5d7c4aad0e2a
Accept: application/octet-stream

EXPECTED RESULT
File - The import status document that would look identical to the response from /import wi

HEADER
Authorization : dc2cff8b-e20d-4e28-8b7e-5d7c4aad0e2a
Content-Type: multipart/form-data

Accept: application/json
passphrase (Optional): password to encrypt the export document

PARAMETER
force_overwrite and environment_id. See the discussion in /import for more detail.

BODY
File - The downloaded export document
```

Expected Result:

```
EXPECTED RESULT
{
"asyncTaskId": 3,
"operation": "IMPORT",
"reference": "IMPORT-AWhwb3J0X2Ru2VtZW50Xzk0Wjlva3JDLmpzb24=",
"status": "RUNNING",
"startTime": "2018-06-16T20:38:31.483+0000",
"cancellable": false
}
```

- On Masking Engine B, retrieve the completed import status using the reference from the returned Async Task response with /file-downloads

```
GET http://masking-engine-A:8282/masking/api/file-downloads/IMPORT-AWhwb3J0X2Ru2VtZW50Xzk0W

HEADER
Authorization : dc2cff8b-e20d-4e28-8b7e-5d7c4aad0e2a
Accept: application/octet-stream
```

Expected Result:
File - The import status document that would look identical to the response from /import with the same export document

Syncing a Masking Job

The following steps provide an example of how to export a Masking Job from Masking Engine A to Masking Engine B using the synchronous endpoints of /export and /import. This presumes that all of the global objects such as algorithms and domains that the masking job relies on have already been synced. This can also be done via the asynchronous endpoint with the same workflow as above.

- On Masking Engine A, export the MASKING_JOB using the /export endpoint.

```
POST http://masking-engine-A:8282/masking/api/export

HEADER
Authorization : dc2cff8b-e20d-4e28-8b7e-5d7c4aad0e2a (whatever you get from login)
Content-Type : application/json
Accept: application/json
passphrase (Optional): password to encrypt the export document

BODY
[
{
"objectIdentifier": {
"id": 4
},
"objectType": "MASKING_JOB"
}
]
```

!!!note
To sync a profile job, swap out the objectType for "PROFILE_JOB" and provide the id of the profile job to sync. Profile jobs are syncable starting in version 5.3.2.0.

Expected Result:


```
{
"exportResponseMetadata": {
"exportHost": "masking-engine-A:8282",
"exportDate": "Fri Jun 15 20:16:20 UTC 2018",
"requestedObjectList": [
{
"objectIdentifier": {
"id": 1
},
"objectType": "MASKING_JOB",
"revisionHash": "579850b1c88baf74cee6bad61d81e2aa3dcc206c"
}
],
"exportedObjectList": [
{
"objectIdentifier": {
"id": 1
},
"objectType": "DATABASE_RULESET",
"revisionHash": "bf63b401129cbc84f90eeb708281e98121f5a829"
},
{
"objectIdentifier": {
"id": "FIRST_NAME"
},
"objectType": "DOMAIN_REFERENCE",
"revisionHash": "e6a52079843afd2625f20237fd50f56254c7e630"
},
{
"objectIdentifier": {
"id": 1
},
"objectType": "MASKING_JOB",
"revisionHash": "579850b1c88baf74cee6bad61d81e2aa3dcc206c"
},
{
"objectIdentifier": {
"id": 1
},
"objectType": "DATABASE_CONNECTOR",
"revisionHash": "6455f39dfa354a54bdf4ef69d6511a6c2bb19db3"
},
{
"objectIdentifier": {
"algorithmName": "FirstNameLookup"
},
"objectType": "ALGORITHM_REFERENCE",
```

```
"revisionHash": "13b0a51a7e3904f52526c442419c54b39033dca3"
}
]
},
"blob":     "RAAAAAokZmZhNWIxNjktODMwMC00N2FlLWJjZmMtNjVhNDUzYWI3OTBjEhgyMDE4LTA2LTE1VDIwOj
"signature": "MCwCFAWGf/97wb+oYuSQizj8U12n7jpQAhQKGCaOJ4U8XyDAOEhMUWkzZXHrpw==",
"publicKey": "MIHxMIGoBgcqhkjOOAQBMIGcAkEA/KaCzo4Syrom78z3EQ5SbbB4sF7ey80etKII864WF64B81uRp
}
```


!!!note
The requestedObjectList returns the list of objects you've requested in the export, and the exportedObjectList returns a list of all objects that were exported. This will include both the requested ones and their dependencies.

- On Masking Engine B, import the masking job. You will need to provide an environment for it to import into.

```
POST http://masking-engine-B:8282/masking/api/import?force_overwrite=false&environment_id=1

HEADER
(same as export)

PARAMETER
force_overwrite and environment_id. See the details in the Masking API Call Concepts sectio

BODY
(Whatever gets returned from export)
```

Expected Result:


```
[
{
"objectIdentifier": {
"id": 3033
},
"importedObjectIdentifier": {
"id": 1
},
"objectType": "DATABASE_CONNECTOR",
"importStatus": "SUCCESS"
},
{
"objectIdentifier": {
"id": 5421
},
```

```
"importedObjectIdentifier": {
"id": 1
},
"objectType": "DATABASE_RULESET",
"importStatus": "SUCCESS"
}
...
]
```