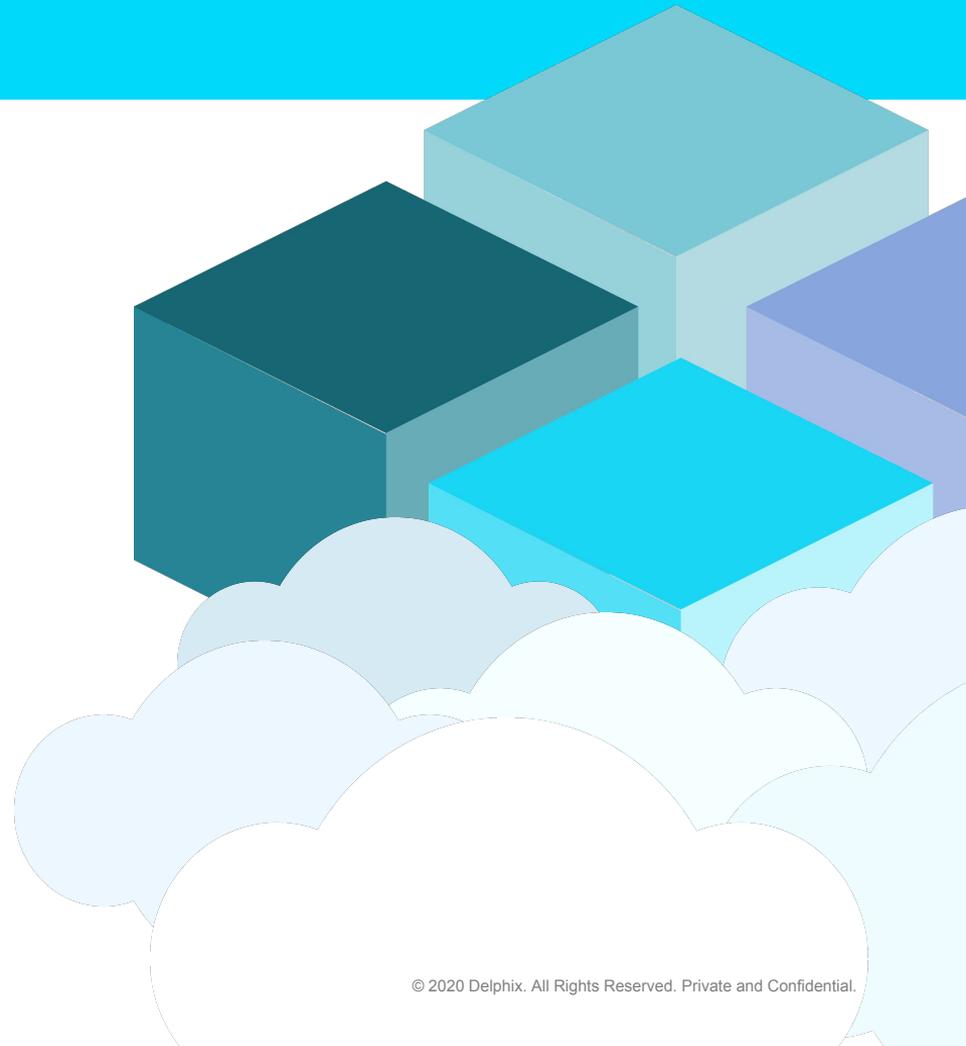


# Delphix 6.0.7 and More

March 2021

Product Marketing



# Delphix Data Platform 6.0.7: **What's New**

- **High Level Updates**

- New Date Masking Frameworks
- New Payment Card Masking Algorithms

- **Data Control Tower Updates**

- Global Masking Users and Groups Engine Changes

- **Technical Updates**

- New Data Masking Frameworks: Date Replacement, Date Shift, Dependent Date Shift
- Virtualization SDK Support for Password Vaults
- ASE Device Mapping Improvements
- Improved Storage Utilization for Large Pools
- PVSCSI Support
- Simplified Connection Management for Oracle DB
- Db2 support for Backup Database Online Exclude Logs

- **Certifications**

- **Deprecations and End of Life**

## **New Date Masking Frameworks**

- We have introduced new date masking frameworks that remove the need for many custom date algorithms.
- New default implementations include date replacement, date shift and multi-column dates (see details in Date Masking Frameworks slides).

## **New Payment Card Masking Frameworks**

- We have added a robust payment-card masking framework as well as a default algorithm for credit card data.

# Masking Engine Changes: Global Object Control

Version 6.0.6.0 of the virtualization engine was designed to integrate smoothly with Users and Groups.

- After enabling the feature in DCT for a 6.0.6.0 engine, user permissions on the engine will become read-only. Changes to be made to permissions on the engine will be synchronized with the engine on a regular cadence (see *Data Versioning* in speaker notes for more detail).
- Masking engines will now adopt the same Users and Groups feature.
- Note that Sysadmins for the setup app are not managed by DCT; they will continue to be managed on the engine.
- Note we support engine versions  $\geq 5.3$ , and need to provide the best experience we can for released versions and also a graceful transition when you upgrade your engines. In versions prior to 6.0.6.0, users can still make changes to permissions on the engine, but DCT is still the source of truth, and will overwrite any changes made on the engine during synchronization.

# New Date Masking Frameworks

**Date Shift:** The Date Shift framework masks date values to different dates based on a specified range around the input value. Masked values are calculated algorithmically using the algorithm's key, so rekeying the algorithm will cause different outputs to be generated for each input. All valid input values will be masked to a new value, and the new value will never match the input.

Configuration via UI

### Select Framework

- Secure Lookup
- Character Mapping
- Payment Card
- Date
- Dependent Date Shift
- Segment Mapping (legacy)
- Mapping
- Binary Lookup
- Tokenization
- Min Max
- Data Cleansing
- Free Text Redaction

### Create Date Algorithm

**Algorithm Name**

**Description**  
This algorithm masks the input to a date that between 5 days before to 5 days after (inclusive). The input will not mask back to the same value even though the value 0 is within the specified range.

**Select Algorithm type** [Learn More](#)

**Replacement:**

**Shift:**

**Roll**

**Unit**

Possible Inputs and Outputs

Input	Output
2016-03-04 12:31:43	2016-03-08 12:31:43
2020-01-02 11:13:45	2019-12-31 11:13:45
1995-12-13 08:00:00	1995-12-14 08:00:00
1998-01-27 06:56:00	1998-01-22 06:56:00

# New Date Masking Frameworks

**Date Replacement:** The Date Replacement framework masks date values to different dates based on specified beginning and end dates. Masked values are calculated algorithmically using the algorithm's key, so rekeying the algorithm will cause different outputs to be generated for each input. It is possible for an input to be masked to the same value.

## Configuration via UI

### Select Framework

- Secure Lookup
- Character Mapping
- Payment Card
- Date
- Dependent Date Shift
- Segment Mapping (legacy)
- Mapping
- Binary Lookup
- Tokenization
- Min Max
- Data Cleansing
- Free Text Redaction

### Create Date Algorithm

**Algorithm Name**

**Description**  
This algorithm masks the input to a date within the range February 1st, 2021 5:00am to February 28th, 2021 2:00pm (inclusive). The input may mask back to the same value.

**Select Algorithm type** [Learn More](#)

**Replacement:**

**Shift:**

**Unit**

## Possible Inputs and Outputs

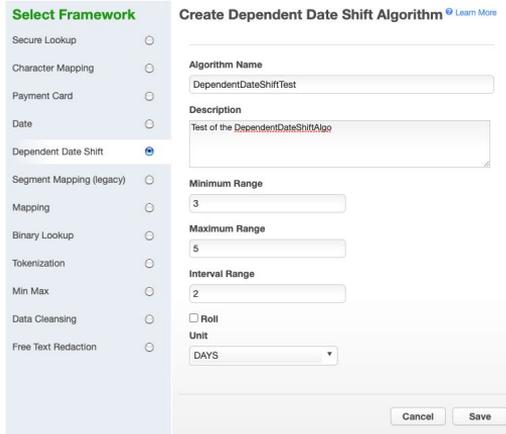
Input	Output
2012-10-04 00:00:00	2021-02-15 23:00:00
1994-12-31 18:32:56	2021-02-04 03:00:00
1994-12-31 19:32:56	2021-02-19 12:00:00
2042-07-13 07:13:42	2021-02-28 14:00:00

# New Date Masking Frameworks

**Dependent Date Shift** : The Dependent Date Shift algorithm takes in 2 dates (designated date1 and date2). It masks date1 based on the provided values for minRange, maxRange, unit and roll. It then modifies the original interval based on intervalRange and unit to calculate date2. If the dates differ but the returned interval is zero (i.e.: the difference between the dates is smaller than the interval value), we assume the interval value to be 1 if date2 is later than date1 and -1 if date1 is later than date2.

The masked results are deterministic for each pair of inputs with the same algorithm key and date and interval ranges. The algorithm does not allow for zero mask so all masked values will never be equal to the input. If date1 is not provided, date2 will be masked based on the provided values for minRange, maxRange, unit and roll.

## Configuration via UI



The screenshot shows a configuration interface for the 'Dependent Date Shift' algorithm. On the left, a sidebar titled 'Select Framework' lists various options: Secure Lookup, Character Mapping, Payment Card, Date, Dependent Date Shift (selected), Segment Mapping (legacy), Mapping, Binary Lookup, Tokenization, Min Max, Data Cleansing, and Free Text Redaction. The main area is titled 'Create Dependent Date Shift Algorithm' and includes a 'Learn More' link. It contains the following fields and controls:

- Algorithm Name:** A text input field containing 'DependentDateShiftTest'.
- Description:** A text area containing 'Test of the DependentDateShiftAlgo'.
- Minimum Range:** A numeric input field containing '3'.
- Maximum Range:** A numeric input field containing '5'.
- Interval Range:** A numeric input field containing '2'.
- Roll:** A checkbox that is currently unchecked.
- Unit:** A dropdown menu currently set to 'DAYS'.
- Buttons:** 'Cancel' and 'Save' buttons at the bottom right.

# New Payment Card Masking Frameworks

The Payment Card framework masks payment card numbers based on the starting digits to be preserved and the minimum number of positions to be masked.

This framework is built on top of the Character Mapping Algorithm Framework with a character set of [0-9]. All characters outside of this character group remain unmasked.

Masked values are calculated algorithmically using the algorithm's key, so rekeying the algorithm will cause different outputs to be generated for each input. The last digit may remain the same if the calculated check digit is equivalent to the last digit of the input. Any inputs with more than one digit will never mask to the original value.

## Credit Card Algorithm Instance

The functionality of the Credit Card algorithm has been changed from the previous built-in version of the credit card algorithm. This instance is implemented in the extensible algorithm framework and is an instance of the Payment Card algorithm framework. The new behavior is consistent with the properties of the Payment Card framework and replaces the previous version. The new version now provides consistent masking results.

In the Algorithms tab on the Masking Engine, Credit card will show the framework as type **PC** (Payment Card) as it is now an instance of the Payment Card Framework.

# Payment Card Example

## Configuration via UI

### Select Framework

- Secure Lookup
- Character Mapping
- Payment Card
- Date
- Dependent Date Shift
- Segment Mapping (legacy)
- Mapping
- Binary Lookup
- Tokenization
- Min Max
- Data Cleansing
- Free Text Redaction

### Create Payment Card Algorithm

**Algorithm Name**

**Description**

**Minimum Masked Positions**

**Preserve Starting Digits**

## Possible Inputs and Outputs

Input	Output
6011458658265198	6011458658118413
6011-4586-5826-5198	6011-4586-5811-8413
3544 4809 7154 9491 806	3544 4809 7143 7633 097
3544.4809.7154.9491.803!	3544.4809.7143.7633.094!

# Virtualization SDK Support for Password Vaults

- Building off of the existing CyberArk and Hashicorp support for Oracle, SQL Server and SAP ASE database user credentials, Delphix Virtualization has extended password vault coverage to the virtualization SDK (vSDK).
- This new feature gives plugins access to the password vaults configured in the engine in a backwards-compatible manner
- The core feature is a new type of property that plugin schemas can use to let the user enter authentication data
- For a schema property of this new type, the user can choose to supply credentials directly (e.g. enter a username and a password) or indirectly by selecting a vault and the parameters to locate the credentials in that vault
- Plugins can also upgrade existing password properties to the new credentials type in their migration scripts
- Replicating plugin objects that reference vaults replicates those vaults to the target engine

UI



# ASE Device Mapping Improvements

- The Delphix Virtualization experience with SAP ASE heavily relies on and mirrors a database's device allocation from the initial load (creating the dSource) to provision (creating VDBs). However, dramatic device layout changes can negatively impact performance.
- This enhancement provides a quality-of-life (QoL) improvement to the overall ASE experience by providing better error handling and escape valves in case a dSource moves into a bad state due to a major device layout shift.
- We have made changes to the exception handling and converted the required DFEs into DUEs/faults.
- These DUEs/faults provide a detailed message about the problem encountered during remap, along with possible causes for the issue and suggested remedies for the same so that user can take appropriate actions on their own without opening a support case.

## CRITICAL

Failed to remap devices from source database to staging database  
Feb 2, 2021 11:30 AM

While performing validated sync using backup taken at "Tue Feb 02 11:12:26 PST 2021" using dump files ["/tmp/ase/corner\_tran3.dmp"], failed to remap source database fragments "

```
[ASEDBDeviceFragmentDO{fragmentType=DATA, pageOffset=0,
sourceDeviceNumber=4, stagingDeviceNumber=0, fragmentSize=2560,
fragmentFreePages=1682}, ASEDBDeviceFragmentDO{fragmentType=DATA,
pageOffset=2560, sourceDeviceNumber=5, stagingDeviceNumber=0,
fragmentSize=2560, fragmentFreePages=2550},
ASEDBDeviceFragmentDO{fragmentType=LOG, pageOffset=5120,
sourceDeviceNumber=7, stagingDeviceNumber=0, fragmentSize=1280,
fragmentFreePages=1275}, ASEDBDeviceFragmentDO{fragmentType=DATA,
pageOffset=6400, sourceDeviceNumber=6, stagingDeviceNumber=0,
fragmentSize=2560, fragmentFreePages=2550}]]" to existing staging
database fragments "[ASEDBDeviceFragmentDO{fragmentType=DATA,
pageOffset=0, sourceDeviceNumber=4, stagingDeviceNumber=0,
fragmentSize=2560, fragmentFreePages=1682},
ASEDBDeviceFragmentDO{fragmentType=DATA, pageOffset=2560,
```

```
8325845d5629-staging-19/datafile/dx9PPtj8vzQ7eectpv_rner_dev5,
deviceSize=2560, totalFragmentSize=2560, totalFragmentFreePages=2550}},
ASEDeviceKey(4)=[ASEDBDeviceDO{sourceDeviceNumber=4,
stagingDeviceNumber=12, deviceName=dxt919mOOzA7cK770L_rner_dev4,
devicePhysicalName=/work/toolkit/564d22b8-b506-a6c7-3a2c-
8325845d5629-staging-19/datafile/dx39luPx0t4b9lvcv5_rner_dev4,
deviceSize=2560, totalFragmentSize=2560, totalFragmentFreePages=1682}},
ASEDeviceKey(7)=[ASEDBDeviceDO{sourceDeviceNumber=7,
stagingDeviceNumber=10, deviceName=dxnubd79uPkAv57r4M_rner_dev7,
devicePhysicalName=/work/toolkit/564d22b8-b506-a6c7-3a2c-
8325845d5629-staging-19/datafile/dxu3hv2Pv7sQ32f7Pn_rner_dev7,
deviceSize=1280, totalFragmentSize=1280,
totalFragmentFreePages=1275}]]".
```

**User Action:** Make sure that the backup being ingested belongs to backup of source database "corner". If the problem persists, perform a SnapSync using a full database backup.

 [Resolving and Ignoring](#)

[Resolve](#) [Ignore](#)

## Improved Storage Utilization for Large Pools

- Up through the 6.0.6 release, Delphix Virtualization has enforced a storage usage limit of 85%. Once met, this limit will cause certain API operations to be disabled to ensure engine data integrity.
- In the 6.0.7 release, this threshold has been relaxed significantly. The new thresholds are as follows:
  - Previously, we'd reserve 15% of the pool regardless of the size of the pool.
  - Now, we reserve 10% of the pool, but with a maximum amount of 1TB.
  - We impose the same limitations as before, when storage utilization begins to consume the reserved space.
- 100TB pools have 16.4% more available space; ~82.28TB before, ~95.80TB now.
- 50TB pools have 15.2% more available space; ~41.14TB before, ~47.40TB now.
- 20TB pools have 11.5% more available space; ~16.46TB before, ~18.36TB now.
- 10TB pools have 5.8% more available space; ~8.23TB before, ~8.71TB now

# PVSCSI Support

- In addition to LSI Logic, Delphix has added support for the VMware Paravirtual vSCSI controller (aka PVSCSI).
- While VMware designed PVSCSI to support very high throughput with minimal processing cost, the performance improvements on Delphix engines can vary from case to case.
- In 6.0.7, we also support manual changes from LSI Logic to PVSCSI for currently deployed engines.
- In our documentation “Customizing the Delphix Engine System Settings” page, you can see how to change back to LSI Parallel if required.

# Simplified Connection Management for Oracle DB

- Prior to this release, we established connections to Oracle databases from two distinct locations (remote connections from the Delphix Engine and local connections from the Delphix toolkit) and we communicated with Oracle databases using two different users (an OS user and a DB user).
- Now all communication with Oracle databases will be performed locally on the database server and connections will be established using OS authentication, thereby eliminating the need for a Delphix DB user.

## Information

Validation will succeed if the OS User Credentials can access the selected database, and the entered Database User Credentials will be ignored. If the OS User Credentials cannot access the database, the Database User Credentials will be used.

**Database Username**

**Database Password**

Validate

## Db2 Support for Backup Database Online Exclude Logs

- Db2 users will now be able to use their online backups without logs on Delphix.
- This feature offers you greater flexibility in how you manage your online backups and logs for Db2.
- Given that logs are no longer packaged with the backup file, you will need to provide them by specifying an alternate path. The logs will then be applied to the dSource during snapshot creation.

# Certifications

- Oracle 19c with RHEL 8.3
- ASE 16.0 with RHEL 8.3

# Deprecations and End of Life

- **Legacy DB Version Support** (Deprecated in 6.0.5, EOL in 6.0.7):
  - Oracle 10 - Reached Oracle End of Extended Support (EoES) July 2013
- **Legacy Hypervisor Version Support** (EOL in 6.0.7):
  - ESX 5.5 - Reached VMware End of Standard Support (EoSS) September 2018
- **Delphix Reporting (Mission Control) EOL:**
  - As of July 2021, Delphix will end-of-life Delphix Reporting.
- **Masking Connectors** (EOL in 6.0.7):
  - Db2 LUW and z/OS v9, Db2 LUW and z/OS v10
  - SQL Server 2005, 2008, 2008 R2