As you may have guessed, static data refers to a fixed data set—or, data that remains the same after it’s collected. Dynamic data, on the other hand, continually changes after it’s recorded in order to maintain its integrity.

Although the concept of dynamic data seems simple enough, making static data more dynamic often seems like a daunting task. Because many businesses aren’t sure how to start the process of data cleanup and diligent data management, they simply don’t do it.

If this sounds like your business, your static data might be hurting you more than you think. Consider these statistics:

40% of business objectives fail due to inaccurate data (source).

Bad data costs U.S. businesses more than $611 billion each year. (source)

64% of “very successful” data-driven marketers say improving data quality is the most challenging obstacle to achieving success (source).

Static

**A large downside to SDM, however, is that it is unable to easily scale when larger data sets and/or combinations of access levels are introduced. Additionally, since the data is static, it is not well suited for analytical use cases. For these reasons, organizations should stay away from SDM for analytical purposes.**

Dynamic

**The downside of dynamic data masking is that stored procedures can’t be dynamically masked. Masking of stored procedures requires rewriting the query results, not the query itself.**

Differences

Static data masking (SDM) permanently replaces sensitive data by altering data at rest within database copies being provisioned to DevOps environments. Dynamic data masking (DDM) aims to temporarily hide or replace sensitive data in transit leaving the original at-rest data intact and unaltered. There are use cases for both solutions, but comparing them as alternative options and/or calling them both ‘masking’ is clearly a misnomer of sorts.

SDM is primarily used to provide high quality (i.e., realistic) data for development and testing of applications within the non-production or DevOps environments, without disclosing sensitive information. Realism, rich data patterns, and high utility of the masked data is critical as it enables end-users to be more effective at conducting tests, completing analytics, and/or identifying defects earlier in the development cycle, therefore driving down costs and increasing overall quality. Leveraging SDM also provides critical input into privacy compliance efforts with standards and regulations such as GDPR, PCI, HIPAA, that require limits on the use of data that identifies individuals. By leveraging SDM, the organization reduces the volume of ‘real’ sensitive data within their overall data landscape, thereby reducing the risks and costs associated with a data breach while simultaneously supporting compliance efforts.

So, SDM clearly has a role in supporting overall data security efforts and in securing the DevOps environment, but what about DDM? How do they compare? Well, as previously mentioned… they don’t.

The reality is, DDM is primarily used to apply role-based (object-level) security for databases or applications in production environments, and as a means to apply this security to (legacy) applications that don’t have a built-in, role-based security model or to enforce separation of duties regarding access. It’s not intended to permanently alter sensitive data values for use in DevOps functions like SDM.

Okay, how does it work? At a high level, sensitive data remains within the reporting database that is queried by an analyst with DDM. All SQL issued by the analyst passes through the database proxy which inspects each packet to determine which user is attempting to access which database objects. The SQL is then modified by the proxy before being issued to the database so that masked data is returned via the proxy to the analyst. To that end, the complexities involved in preventing masked data from being written back to the database essentially mean DDM should really only be applied in read-only contexts such as reporting or customer service inquiry functions.

Adv of SDM as compared to DDM

First, it’s complex and complicated. It’s not as simple as installing a software application and running it. The organization must undertake a detailed mapping of applications, users, database objects and access rights required to configure masking rules; and maintaining this matrix of configuration data requires significant effort.

Second, it can be risky. Some organizations we’ve worked with are hesitant to adopt DDM given the inherent risk of corruption or adverse production performance. In addition, relative to SDM, DDM is a **less mature technology** for which customer success stories are not as well known and use-cases are still being defined.

Finally, the fact remains that the underlying production values and sensitive fields are not actually de-identified or masked, meaning the risk of exposure remains; particularly if the organization in question is leveraging this data to provision to DevOps without a static masking solution in place. So, if your goal is to increase data security efforts relative to data breach risks and/or compliance support efforts, you’re no further ahead with dynamic masking.

All this is not intended to suggest DDM does not have a role in data security, or that it is not as effective as SDM. The point is that they are two fundamentally different solutions operating in differing environments, for varying purposes. The key question organizations must ask is simple: what is the business problem or data security challenge we are trying to solve? That question will help determine which solution makes the most sense.

At the end of the day, however, data de-identification via static data masking is a data security solution recommended by industry analysts as a must-have protection layer in reducing your data risk footprint and the risk of breach by inside or outside threats. Dynamic data masking acts more like a role-based access security layer within the production environment and for internal user privilege requirements- and there are options available within other solutions to achieve this end-goal.