syncsort

Encryption, Tokenization, and Anonymization for IBM i

A Quick Guide to Protecting Sensitive Data

Introduction

Over the years, high-profile security breaches have spawned new and ever-expanding compliance regulations. These regulations are forcing companies to increase measures to protect sensitive data, especially personally identifiable information, to prevent it from being seen by unauthorized parties. This not only applies to the activity of hackers. It applies equally to internal staff, contractors, and business partners, all of whom should be able to view only the information they are required to access in order to do their jobs.

Three major solutions that are effective for keeping sensitive data out of the view of prying eyes are encryption, tokenization, and anonymization. This guide will help you gain a high-level understanding of each, learn how the underlying technologies work, and recognize the situations in which companies typically use these solutions.

Encryption

The method that organizations have used the longest to protect sensitive data is encryption, which combines one or more publicly available algorithms with a specialized, secret piece of data called an encryption key. Together, the algorithm and the encryption key transform human-readable information (sometimes referred to as plain text) into an unreadable format (sometimes referred to as cipher text). When the data needs to be decrypted to return it to its original form, it is done through the use of the same encryption key that was used when the data was originally encrypted.

Encryption can be used to protect data at rest or data in motion. The data at rest on IBM i that is a candidate for encryption includes any sensitive data stored on the system, whether that data resides in specific fields within a database file, across entire databases, in save files, in spooled files, or on backup tapes. Data in motion refers to data that is transmitted over a network from one system to another, or one entity to another. This may be done using SFTP (SSH Secure File Transfer Protocol), FTPS (FTP over SSL), encrypted Telnet, or HTTPS. Regardless of whether data is to be encrypted at rest or in motion, you will need to obtain or create a solution that provides the specific type of encryption you require. The solution must include implementation of an approved encryption algorithm such as Advanced Encryption Standard (AES), generation of strong encryption keys, and protection of encryption keys by a key management solution.

It is important to note that it's critical to use algorithms that meet current standards from either the U.S. National Institute of Standards and Technology (NIST) or from other internationally recognized standards organizations. Older algorithms that don't meet current standards are more likely to be compromised by hacker attacks. The use of current encryption algorithms and the proper management of encryption keys are critical to ensuring that your encryption solution reliably protects sensitive data.

Keys to the Kingdom

Encryption algorithms are publicly available; however, the keys your encryption solution uses to encrypt and decrypt data are yours alone. This is why it is critical to implement sound methods for creating, distributing, and storing your keys.

Best practice in key management is to store keys on a different server than the one containing the data encrypted with those keys. On IBM i for example, the keys might be stored on a separate machine or partition. Third-party key management solutions are available that handle the management of encryption keys as a "lifecycle" in which the creation, activation, usage, rotation, expiration, retirement, and destruction of encryption keys is carefully controlled to ensure that keys are used only for limited periods of time.

Many compliance regulations, such as PCI DSS, require the implementation of strong key management practices. Those practices include checks and balances such as separation of duties as well as dual-control processes that require at least two or more individuals to be involved when key management functions are implemented or changed.

Field Procedures for IBM i

In years past, implementing field-level encryption on IBM i required applications to be changed to use encryption APIs. However, beginning with V7R1 of the operating system, field-level encryption became significantly easier to implement due to the addition of a column- or field-level exit point called FieldProc. With FieldProc, application changes are often not required.

As with other types of exit points, IBM provides the architecture in the OS for the exit point that is used to invoke a user-created exit program, but typically IBM does not provide the exit program itself. Some shops might create their own encryption application based on the documented architecture of the FieldProc exit point while many choose to use an encryption solution from a third-party vendor such as Syncsort. Either way, the encryption solution must utilize exit programs designed for FieldProc. To learn more about using FieldProc to implement encryption within applications, click here to download the Syncsort e-book *IBM i Encryption with FieldProc.*

Field Encryption and the Use of Masking

When field-level data is decrypted for use in an application, report, etc., you may want users to see all, a portion, or none of the decrypted field value, depending on the privileges of each user. When all or part of the data is hidden (masked), each character of the sensitive data is replaced with a value such as an "X" or an asterisk. Many third-party encryption solutions, including those from Syncsort, provide masking functionality.

Encryption Pros

- Encryption is a mature technology with a recognized body of standards, independent certification of vendor technologies, and continual scrutiny of algorithms by international standards organizations and the larger cryptographic community. Organizations that deploy independently certified encryption solutions enjoy a high level of confidence in the protection of their data assets as well as confidence in meeting requirements of compliance regulations that mandate the protection of sensitive data.
- Encryption of sensitive data is a requirement of several compliance regulations, including GDPR and PCI DSS. Some regulations have provisions that waive public notification requirements should a breach occur, but only if the encryption algorithms and key management practices comply with NIST or similarly accepted standards.

Encryption Cons

- Depending on the implementation of the encryption solution, the process of encrypting and decrypting field data may negatively affect CPU utilization; however, advanced encryption solutions like those from Syncsort have only a negligible impact on performance.
- In some cases, encryption may not preserve the original format of fields, which can affect field-validation processes.
- Encryption can affect indexes used for searching and sorting of data in legacy RPG applications, which may require application changes. With modern SQL applications, this is not the case. Nonetheless, some encryption solutions, like those from Syncsort, provide add-ons that make it possible to use encrypted indexes with legacy RPG applications without having to make changes to the applications.
- Numeric database fields can present a challenge for encryption. The act of encrypting a numeric field could make it invalid in applications, thus causing database errors. For this reason, many IT shops convert numeric fields to character or binary format before starting their encryption project.

A note about encryption cons: Because third-party encryption solutions, like those from Syncsort, can mitigate cons such as performance impact or the need to make changes to legacy RPG applications to allow for the use of encrypted indexes, many shops choose to buy rather than build a solution. The same is true for key management features. Unless a shop has an in-house cryptography expert, it is preferable to look at a third-party solution that provides full key management lifecycle capabilities and protects keys from unauthorized access.

Tokenization

A different approach to shielding sensitive data is to replace this data with non-sensitive substitute values called "tokens." Often used to replace credit card numbers, social security numbers, and other personally identifiable data in applications and reports, tokenization utilizes a database—sometimes referred to as a token vault—to store the information about the relationship between the sensitive data and its replacement token. This makes it possible to retrieve the sensitive data whenever necessary. For maximum security, the token vault is kept on a separate, encryption-secured server, with all transfers of data taking place using encrypted communications.

Because tokenization separates sensitive data from production databases via the token vault, it reduces the risk of this data being exposed should the production database be breached. Because tokens have no algorithmic relationship to the original data, a stolen token can never be "cracked" to obtain the original value. This approach has significant benefits when it comes to meeting compliance regulations because the production server, on which the sensitive data is tokenized, won't be required to demonstrate compliance to auditors since no sensitive data is kept on that server.

Tokens are often used by retailers to represent credit card numbers. The actual credit card numbers are stored on a credit card payment network rather than on the retailer's systems. The retailer's systems keep only a tokenized version of each credit card number.

When the original data needs to be retrieved and displayed, many tokenization solutions allow masking of all or a portion of the value with a character such as an "X" or an asterisk, with the number of characters hidden based on the user's privileges.

In addition to protecting sensitive data in production environments, tokenization is often used in development, testing, business intelligence, and other non-production environments. However, in these types of environments, unrecoverable tokens are used, which removes all connection to the original data. More about this approach in the upcoming section on anonymization.

There are cases when tokens are required to match the characteristics of the original data in order to preserve the integrity of applications and databases. This is called format-preserving tokenization, and this approach typically substitutes a similarly formatted value for the original value. For example, social security numbers would be replaced with tokens that still look like social security numbers. Token consistency can also be ensured so that the same token is used for each instance of the original data wherever the data appears in the database; for example, each instance of driver's license number 9353036 could always be replaced by 4659003.

Tokenization Pros

- Tokens have no algorithmic relationship to the real data they represent in the way encryption does. Because of this, if tokenized data is breached, no decryption key could reverse the tokens back to the original data values.
- Sensitive data is completely removed from the production system so that, if the system is breached, there is no way for a hacker to obtain the sensitive data.
- With sensitive data removed from a server through tokenization, the server is removed from the scope of compliance. Not only does this remove the exposure of the server from possible breach, it could save a company money by potentially eliminating the need to audit these servers in order to meet compliance regulations.
- The original format of sensitive data fields is retained, and indexes are not affected.
- The need to manage encryption keys is eliminated.

Tokenization Cons

- Tokenization does not enjoy the same level of recognition as encryption among international standards bodies such as NIST, nor within many compliance regulations.
- Tokenization has a greater impact on performance than encryption due to the actions that necessarily occur between the production database and the token vault whenever tokenized data is added, changed, or deleted or when it is necessary to retrieve the sensitive data from the token vault. The performance of the token vault can be further slowed in high-transaction situations when the tokenized data needs to be encrypted/decrypted as it is being used, which is typically a requirement.

Anonymization

There are instances in which a company may want to permanently replace sensitive data with a substitute value. This is accomplished with anonymization technology, which is a type of tokenization that eliminates the use of a token vault. Without a token vault, the original data associated with the token is unrecoverable.

Anonymization is generally not used in production environments. Instead, it is often used for development or test environments as well as to send reporting data to external partners or agencies who are not authorized to access sensitive data. Depending upon the solution used, different methods of anonymizing the data may be offered. For example, one method is to scramble the original data in a way that produces a format-preserving token that remains valid for purposes of the application. In other words, an anonymized credit card number would still show the same number of digits as a credit card and could even be validated by processes that distinguish valid numbers from mistyped or otherwise incorrect numbers. Also, the same data can be anonymized with the same replacement value wherever it is encountered; for example, every instance of Robert Q. Anderson in a database is replaced by Jonathan P. Smith.

Some anonymization solutions integrate with the replication technology of a high availability or disaster recovery solution so that the anonymized environment (e.g., a development environment, a business intelligence environment, etc.) is fed in real time with new data whose sensitive fields have been replaced by non-recoverable tokens.

Anonymization Pros and Cons

Anonymization shares many of the same pros and cons as tokenization, but with a couple of exceptions:

- Pro: Anonymization doesn't have the performance impact of tokenization, which uses recoverable tokens. That's because no external token vault is utilized.
- Con: Anonymization is not typically used for production environments.

Syncsort Can Help

Encryption, tokenization, and anonymization each have their own uses and benefits, and many organizations will choose to implement one or more of these technologies depending on the environment that needs to be protected. As with any other IT security decision, choosing the right data-privacy solution takes a thorough evaluation of your own requirements and the available technology options by your security administrator, compliance team, and management. It also helps to bring in outside experts who understand the technologies, including the potential pitfalls. Syncsort brings you such a team of experts, all with an in-depth knowledge of encryption, tokenization, and anonymization. In addition, our Syncsort Assure portfolio of IBM i security products includes a range of industry-leading encryption, tokenization, and anonymization software, making it easy to find the approach that best meets your company's specific requirements.

Learn more at www.syncsort.com

About Syncsort

Syncsort is the global leader in Big Iron to Big Data software. We organize data everywhere to keep the world working – the same data that powers machine learning, AI and predictive analytics. We use our decades of experience so that more than 7,000 customers, including 84 of the Fortune 100, can quickly extract value from their critical data anytime, anywhere. Our products provide a simple way to optimize, assure, integrate, and advance data, helping to solve for the present and prepare for the future. Learn more at syncsort.com.

www.syncsort.com