# The Difference Between Format-Preserving Encryption and Tokenization

**Format-preserving encryption (FPE) and tokenization have similar objectives, but the difference between the two can be confusing at times.**

**This fact sheet provides a brief overview of both the similarities and differences between these two data-centric security methods and provides examples of when one method might be used instead of the other.**

# Tokenization vs Format–Preserving Encryption

**When it comes to sensitive data, a stateless, data-centric approach is proven to have fewer security gaps and risks, as security travels with the data while it's at rest, in use, and in transit, rather than relying on additional security methods to provide protection if and when the data leaves the application, database, or storage resource.**

## Similarities

**Protection of sensitive data**
In the broadest sense, both tokenization and FPE are data security tools that focus on protecting sensitive data, such as primary account numbers (PANs) or personally identifiable information (PII), from theft or exposure. This is done to minimize the effects of data breaches and to achieve compliance with key requirements of data privacy laws such as GDPR and CCPA or with industry standards such as PCI DSS.

**Data-centric security**
FPE and tokenization also share the same "data-centric" approach to security, meaning data is protected at the element or field level, rather than protecting whole files, data-bases, cloud instances, or storage resources. Given the almost inevitability of breaches and accidental exposure, data-centric security protects the data itself so that it's still protected in case of theft, loss, or unauthorized access.

**Referential Integrity**
Another similarity is that both techniques maintain "referential integrity" meaning that the protected data and the source data have a one-to-one relationship. The analytic value of the source data is maintained by the protected data which is especially advantageous for big data analytics and similar initiatives where data is shared. This feature allows data to stay protected for more than 90% of its lifespan with a mantra of "protect once, unprotect sparingly."

**Reversible anonymization**
FPE and tokenization are also a form of reversible anonymization (also referred to as pseudo-anonymization) meaning that the protected data can be selectively unprotected at certain points where the source data is absolutely required, such as credit card settlement, submitting government forms with ID numbers, EFT transactions, and the like. If need be, implementation strategies can be used to make both technologies irreversible.

**Stateful or stateless implementation**
Both technologies can be implemented in either a stateful or stateless approach. Stateful means that a database is used to track encryption keys or tokens that have been generated and used. This database must be shared (replicated) and will increase in size as more keys or tokens are generated. Stateless means that an algorithm is used to generate a static table of highly randomized values (typically something called an unbalanced Feistel network) that is used to derive encryption keys or tokens. This table does not grow in size or change, which alleviates the scaling and concurrency concerns surrounding stateful implementations, which can be prone to failure and even data loss. In a stateless key management or tokenization approach, any key or token that is generated at any point in time can be re-generated (derived) again with the static table.

## Differences

**Unique to tokenization**
Tokenization generates random data values, commonly referred to as "tokens," which represent actual data. This process typically replaces sensitive data elements with non-sensitive data elements – tokens – of no exploitable value. Tokens usually preserve the same length, format, and composition as the original data to facilitate using tokens without requiring database or application changes and maintaining referential integrity. Tokens are generated by a centralized, stateless token server, so unlike FPE, there is no encryption key that requires management or rotation.
Furthermore, since tokenization is a centralized service, the solution must be designed with fault tolerance, scaling, and failover taken into account. Once generated, tokens can be used indefinitely without the need to retokenize.

**Unique to format-preserving encryption**
Encryption uses an algorithm and a centrally-managed encryption key to encrypt the original data into a similarly protected form. FPE refers to encrypting data in such a way that the output is in the same format as the original data. FPE, like any encryption operation, requires an encryption key to be delivered to the endpoint wherever encryption (or decryption) is performed. In order to maintain referential integrity across datasets, the same encryption key must be used everywhere a data type is found (e.g. all SSNs across the enterprise are protected with the same key).

Since encryption key delivery is an expensive operation, keys are also typically cached for reuse outside the protected confines of the key manager. If the encryption key were to be obtained or guessed by an attacker, then any data protected with that key could be potentially compromised, requiring re-encryption of data. Due to this inherent risk, encryption keys must also be periodically rotated, typically on an annual basis at least, which also requires re-encryption to maintain referential integrity.

Re-encryption of data can be a time-consuming and risky task which must be done securely since the data is vulnerable between the time it is decrypted with the old key and encrypted with the new key. Also, since FPE is almost always implemented with stateless key management, it's impossible to destroy a key because, by definition, it can always be derived on demand. That means that after a key rotation, any data that is not located and re-encrypted will continue to be vulnerable.

comforte

## Side by Side Comparison

**Tokenization has advantages over FPE in most use cases because it reduces complexity and management requirements. Here's quick breakdown of the differences:**

| Operational concern | Tokenization | FPE |
|---|---|---|
| **Avoid re-coding applications and re-structuring databases?** | YES – the original data is replaced with a token, which retains the format of the original data. | YES – the original data is encrypted and formatted in such a way that the format of the original data is retained. |
| **Can be implemented in a stateless fashion?** | YES – stateless tokenization is ideal since the token server doesn't replicate tokens across its nodes and doesn't store any sensitive data ever. | YES – stateless key management allows for any key to be derived at any point in time and alleviates key replication issues. Stateless keys cannot be destroyed. |
| **Unbreakable?** | YES – hackers cannot reverse engineer tokenized data (or vice versa) as secure, random data was used to generate the tokens. | NO – the systematic encoding process is reversible with the right encryption key or brute force. Some FPE standards have been found to be insecure. |
| **Successfully remove sensitive data?** | YES – because tokenization 'replaces' the original data with a token, therefore the original data no longer exists. | NO – the actual data is still there, it's just scrambled in a reversible pattern, so technically, it's not removed. |
| **Helps reduce overall compliance burden?** | YES - tokenization reduces compliance scope, as compliance auditing affects systems hosting sensitive data. | NO – although FPE fulfills requirements of data protection regulations, the systems still need to be audited, thus the burden of proof is still there. |
| **Remove or reduce the burden of key management?** | YES – tokens are generated by a centralized server which doesn't require rotation or management of encryption keys. | NO – organizations need to rotate encryption keys on an annual basis, which adds to the operational management burden IT departments already face. |
| **Protects data even when user/ admin credentials are leaked?** | YES – tokens are still usable in their protected state so users don't need to have access to unprotected data. | YES – as long as access to the key manager is not compromised, data is protected. |
| **Standards-based approach?** | YES – the ANSI X9.119-2 standard governs the secure generation of tokens. Not all solutions implement the ANSI standard. | YES – the AES-FFx standard governs FPE.<br>Note that currently only AES-FF1 is considered secure as FF2 and FF3 have known vulnerabilities. |

Learn more about how enterprise data protection can benefit your organization here or contact us to discuss your data protection requirements:
www.comforte.com

Follow us on social media: