International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 10 | Oct 2019 www.irjet.net p-ISSN: 2395-0072

Improved Vault based Tokenization to Boost Vault Lookup Performance

Ashish Thakur[1], Amit Saxena[2]

[1]Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal
[2]Truba Institute of Engineering and Information Technology, Bhopal, M.P, India

---

Abstract - *Organization/institutions that deals with sensitive information set of customer such as PCI(payment card industry), PHI(protected health information), and PII(personal identifying information) needs to comply with international as well domestic/local governing regulations like PCI-DSS, HIPAA, and so on of different sectors like finance, healthcare, and insurance. Sensitive data floats between various entities/systems such as payment, customer care, collection, risk and analytics. Until recently, many existing solution were inclined to protecting data at-rest through encryption. However, increase in the incidents of data breaches that target the encrypted sensitive information resulted in taking a new approach termed as tokenization for not only for storage i.e. data at rest as well during transmission. Tokenization - a process where sensitive data or information is substituted with a surrogate value which is a sequence of characters that retains all the vital information without compromising the security of the sensitive information. Unlike encryption, a process where sensitive information is encrypted at source and decrypted at the destination, tokenization substitute sensitive information/data with a stand in token.*

Due to random assignment of the tokens, it is impossible to re-engineer a token to retrieve the original sensitive data. Tokenization utilizes a database (also termed as "vaults") to store the mapping of sensitive information with the substituted values name as "token". With growing database/vaults challenges of latency and scalability arises to protect/secure the sensitive information or data. To overcome such challenges, a solution has been implemented using a distributed database partitioned based on token ranges without the need of syncing the database content and also a tokenization engine taking care of uniformly filling the databases, thereby reducing the latency and increasing the scalability. Unlike vaultless tokenization approach that compromises with basic philosophy of tokenization and uses an algorithm to generate the tokens, where original sensitive data runs through the algorithm as input. The token generated using vaultless tokenization approach are reversible.

Key Words: Data security, Tokenization, Vault tokenization, Vault less tokenization.

1. INTRODUCTION

With the advancement in the technology with the time, many cases of data breaches are being reported every year. As the attackers or hackers and their tools are becoming more sophisticated, one can expect such data breach incident to be growing in years to come. The major responsibility lies with the targeted institutions - banks, financial corporations, healthcare & government bodies - to take the required measures so as to ensure the protection of customer information safe, private and secure. As the threats are growing exponential in the domain of data security both from malicious intrudes as well as non-malicious attackers, it's a high time for enterprises to dealt with this data security challenge on priority. Increasing sophisticated technology and environment, the answer to the dealt with the data breach challenge is not just to add an additional layer or cover the endpoints or the network with traditional encryption methodology.

Rather, it's time to focus on the best possible available solutions to protect vulnerable data, one of which is tokenization. The basic idea behind tokenization based service model is to replace/substitute thee business organization sensitive and confidential data with unique identification pattern to build it secure from attackers/hackers to steal the confidential & sensitive data, so that it fulfil the compliance requirement of the appropriate authority.

There are two kinds of servers currently popular for implementing tokenization

- ✓ Vault based
- ✓ Vault-less

Many data protection companies store their customer sensitive data in a secure database termed as data vault. These data vaults safe house data/information until it needs to be retrieved to make payments, identify the cardholder or to serve other purposes. In case of tokenization, these data vaults are also referred as token vaults, and they can either be on-premise or cloud based. The term vault based means both sensitive data(like credit card number) and token will be mapped & stored in a table usually known as "Teradata" tables. Handling of these table becomes a big challenge with increase in the volume of

transactions. Every time during tokenization it stores a record for each card data and its token. When you used a card multiple times, each time it generates multiple tokens. It is a fundamental concept. So the challenge here is Table size will be increased. In this project work a solution is presented where either on-premise or cloud based tokenization based service model is to protect sensitive and confidential data using vault token with high performance.

The recent technology "Vaultless token" is much faster than the vault based model, where the card numbers and tokens will not be stored. There are two parts of the Vaultless tokenization model - One part deals with the received request to generate the a random number - token, whereas the other part which is the de-tokenization, deals with decoding of the generated random number and send back the actual card number to the requestor. Tokens can be a combination of Alpha-numeric characters. Although Vaultless will not be the appropriate solution for all customers, in the right use cases. This approach also violates the fundamental principle of tokenization which states that - token is a meaningless value & cannot be re-engineered to generate back the original data.

1.1 What is Tokenization?

Tokenization helps in protecting the sensitive data by replacing it with a non-meaningful or non-sensitive data. Tokenization generates an unidentifiable tokenized form of the original data, thereby preserving the format of the source data. For example, a Primary Account Number (9876-1234-4567-5678) when tokenized (1234-5678-9101-1213) looks similar to the original data i.e. PAN and therefore can be used during the financial transaction life cycle , the way the original data(PAN) is used without the risk of linking it to the cardholder personal information. Since the tokenized data also preserves the size and format of that of original data, therefore no changes are required to store the tokenized data in the database schema.

Data tokenization helps controlling and maintaining the compliances during the transaction data flow across environments. Tokenization is not appropriate for the data type like text files, PDFs, MP3s etc., instead file system level encryption is appropriate for such data. The encryption changes the original block f data into an encrypted version of data.

1.2 How does Tokenization works?

Token is the heart of tokenization. In simple terms, a token is a piece of data that stands or substituted for another, more valuable piece of data. Tokens have logically no value on their own – they are only useful or important because they represent something valuable. A good example is a poker chip. As we understand, tokenization works by substituting the valuable sensitive data from the application environment, with the tokens. Most of the business uses some or the other form of sensitive data within their application systems. Data can be a credit card data of a cardholder, Medical information of an individual, Social Security Numbers(SSN), or anything that has to be protected & used securely over the network or environment. Using tokenization, such data is taken entirely out of the application system or environment thereby mitigating the risk of being accessed unlawfully by malicious or non-malicious users. The data is then being replaced with tokens - that are unique set of information. A token in itself has no extrinsic or exploitable value, therefore tokenization is becoming increasingly popular these days considering the data breaches incidents.

As explained, “Even if a malicious attacker or intruder get hold of tokenized data, they in-true sense haven’t gotten anything vital but just a meaningless or non-valued token.” Additionally, tokenization also helps organization to reduce their compliance obligations. The commonly known reason for organizations to choose tokenization over other alternative technology is cost reduction: reduced costs for application changes, followed by reduced audit/assessment scope. The organizations also choose tokenization whenever there is need to upgrade the security for large enterprise applications — as far as lot of changes to be made, tokenization also reduces potential data exposure and thereby minimizing the need for encryption at the same time.

1.3 Tokenization in Action:

What does a **token** look like? There are two types of tokens: Formats preserving and Non-Format preserving. Format preserving tokens preserves the format of the original card number data which is a 16 digit value, this generated tokens resembles to that of original card number. For example:

Payment Card Number: 5236 1111 8597 1163
Format Preserving Token: 4121 8675 2543 1311

Non-format preserving tokens do not resemble the original data at all & therefore looks dissimilar to that of original data and could include both alpha and numeric characters. For example:

Payment Card Number: 5236 1111 8597 1163
Non-format Preserving Token: 25c92e17-f068-415f-d965-7593a32u0322

Most of the organizations, prefers to use a format preserving tokens to avoid additional overhead of data validation issues with existing systems or applications and business processes.

1.4 How exactly do tokens work?

Following steps are involved during the payment transaction done via credit card:

- In the first steps cardholder swipes the credit card on POS machine or the credit card details are entered manually onto an ecommerce website or mobile app.
- The POS machine or e-commerce/m-commerce site sends the Primary Account Number details to the credit card tokenization system.
- The tokenization system then generates a token which is sequence of 16 random characters, that replaces the actual Primary Account Number. Mapping of token along with the Primary Account Number is stored and maintained in a database called as token Vault. The tokenization system also retrieves the associated token and records that are correlated in the token vault.
- The token is returned to the POS terminal or commerce site and represents the customer's credit card in the system.
- If the organization is using a payment processor's tokenization solution, then the token is sent to the payment processor. They use the same tokenization technology to de-tokenize (i.e. retrieve the original sensitive data based on generated token) and view the original credit card number and process payment. If the organization is using a third-party tokenization solution, then the token goes to the third party. The Third party token service provider then de-tokenize and send it to the payment processor for credit card processing.

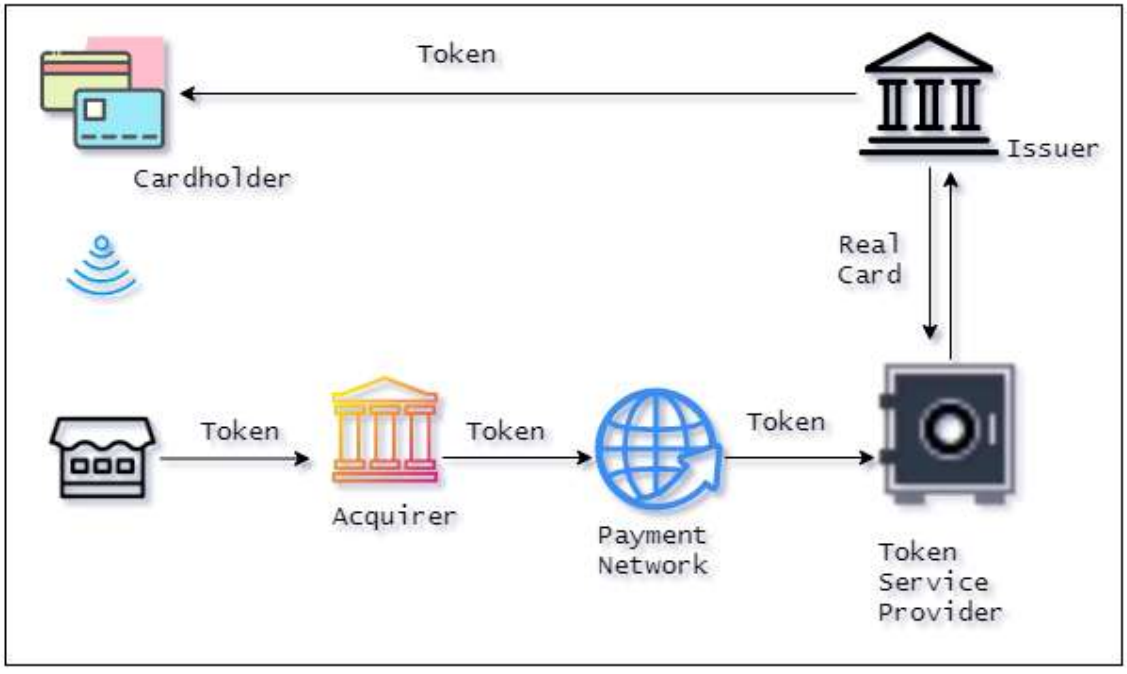

Fig-1: Role of Token Service Provider in Payment Network

2. PROPOSED SOLUTION & RESULT

Tokenization technology is not limited to credit card data and it can be used for any kind of data: social security number, UAID, financial identification number, account number etc. For security reasons tokens are generated by using random number functions, but can also preserve the length, data type of the original sensitive data, to ensure proper integration with other

systems. The original data can be recovered using de-tokenization if required. Tokenization has advantage over encryption thereby minimizing investments, reduced complexity, minimizing administrative burden, reducing the impact on the business processes & enhancing collaboration.

There were some challenges or issues with vault based tokenization, in this project work attempt has been made to address the following challenges of vault based tokenization:

- The vault is a database that keeps mapping of sensitive information to tokens. On a large scale where lot of sensitive data is to be stored, the database grows exponentially, thereby increasing the time required for lookup during search process, and limits the tokenization application performance and also results in increased back-up procedures.
- In addition, the data vault maintenance work also increases whenever new data added.
- This also pose another challenge of ensuring consistency across databases for continuous synchronization of token databases.

2.1 Tokenization: Basic architecture model

Basic Architecture model of tokenization flow:

1. Initially the application collects or generates sensitive data.

2. The sensitive data is being then passed immediately to the tokenization server - point to here that sensitive data is not store locally at application.

3. In the next step, tokenization server generates the random token. The sensitive data is then stored in a secured, restricted, protected database(usually encrypted).

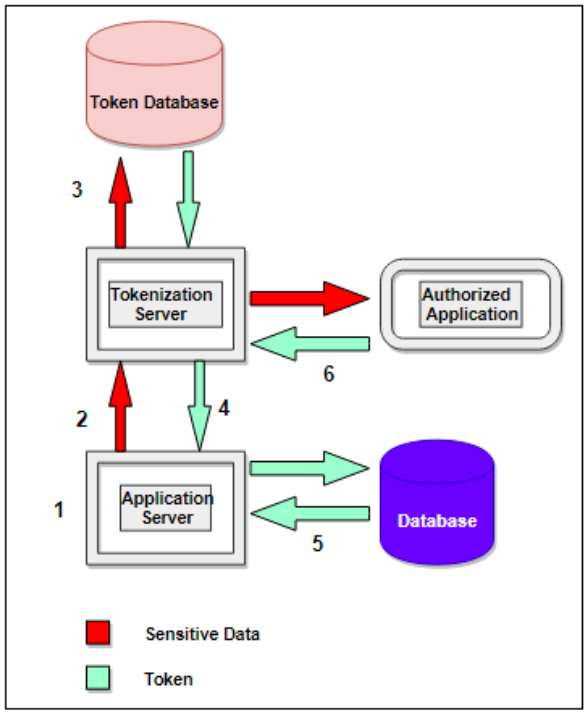

Fig-2: Basic Tokenization Architecture

4. The token generated by tokenization server is returned to the application.

5. The application then stores the token rather than the original sensitive data. The token is then used for transactions with the application.

6. Authorized application or user can only request for the sensitive information using the token. The value is never stored locally, which dramatically limits the potential exposure to attackers.

2.2 Token Servers:

Tokenization server not only does job of creating tokens but, also perform some other functions like encrypting the data, validate the users and returns the sensitive as necessary for authorized access. Tokenization server's design has significant effects on the scalability, performance, accessibility and security of the overall application. Most of the services are completely invisible to end user. Therefore assessment of these functions during designing is equally vital to ensure that we don't run into problems in the later stages. Two basic type of service requests are performed by tokenization server:

- Tokenization server accepts sensitive data like PIIS, credit card number from authorized source, and then respond the calling application with the new or existing token. It also stores the token along with the encrypted original data while creating the new tokens.

- Token server return decrypted original data to the legitimate or authorized application whenever requested with a valid token.

2.3 Architectural enhancement to address the performance challenge of vault token:

Before delving into the nitty-gritty of the technical implementation details of the enhancement done to address the performance challenge as stated in this report in the beginning , let have an insight on understanding the performance latency during the lookup on the token vault or databases, which grows exponentially with addition of token-original data pair into the vault or database. With the growing trend of using the tokenization as a data protection mechanism , many organization/institutions have started upgrading their application to adapt to tokenization for protecting organization or customer sensitive information specially in financial domain. This has given a push for application's operating on large customer databases, thereby posing the latency concerns during token vault lookup that take place whenever request arrived at application for retrieval of the original data value based on token.

Following are the architectural considerations taken into account:

- The token vault or database shall be spitted based on the token range into multiple instance of the vault or database on to multiple site to boost the performance during the lookup.
- Tokenization server acts as a central hosted entity that will be responsible for filling the vault or database placed on multiple sites uniformly.
- Request for creation of token, retrieval of original data from token vault/database, will be available as REST API so that the tokenization server can be integrated with any existing or new applications.
- With the advancement of modern technology & growing trend of cloud computation or infrastructure, there is need for every new application to be easily deployable on cloud platform. Since the proposal here is to make tokenization as a REST API based service, therefore the overall tokenization solution can easily be deployed on cloud platform.

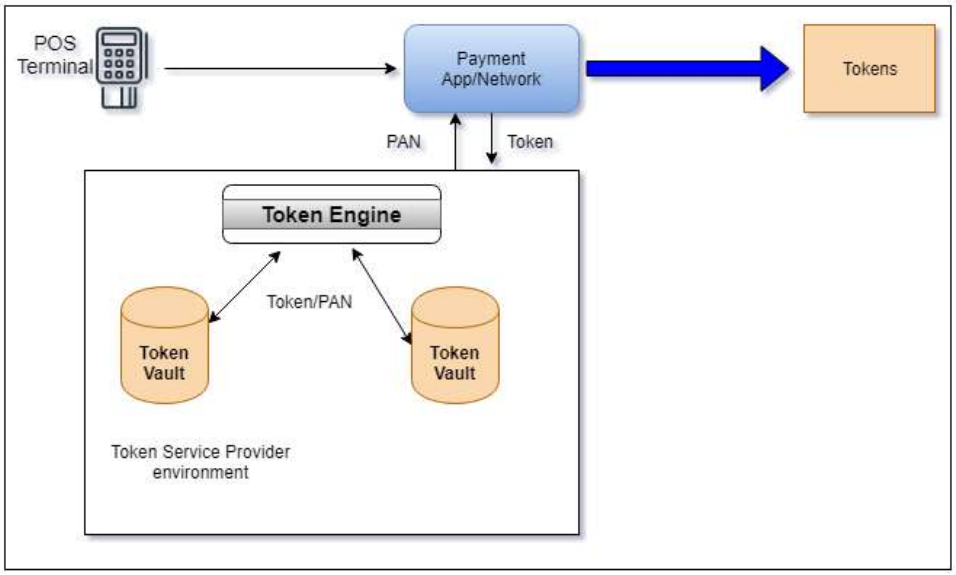

Fig-3: Improved performance of Vault based Tokenization

2.4 RESULT:

Following REST APIs are proposed: Creation of Token, Retrieval of Original sensitive data based on Token

Create a Token API:

Rest API to turn a credit card number into a token using this service. This API takes card details and generate the token representing the card details.

REQUEST PARAMETER			JSON:
Field	Type	Description	
merchant_id	String	Merchant ID which represents the merchant storing the card	
card_number	String	A valid card number	
card_exp_year	String	Expiry year of the card (Format: yyyy) Example: 2019	
card_exp_month	String	Expiry month of the card (Format: mm) Example: 09	
card_security_code	String	The CVV number of the card.	
name_on_card	String	Cardholder name	

{
"card_number":"4567890243749490",
"card_exp_year":"2019",
"card_exp_month":"09",
"card_security_code":"123",
"merchant_id":"VISA",
"name_on_card":"XYZ"
}

RESPONSE PARAMETER			JSON:
Field	Type	Description	
card_token	String	Unique token which represents the card that has been added.Payment transaction can be initiated with this token.	{ "error":"", "reference_number":"15102913382030662954", "success":true, "card_token":"546454597415455" }
Error	String	If something failed	
reference_number	String	Unique reference for a transaction.	
Success	bool	True/False	

API to Retrieve the original data using Token:

To get the details of the store token of a customer, simple make a request with token .

REQUEST PARAMETER			JSON:
Field	Type	Description	
card_token	String	Unique token which represents the card that has been added.Payment transaction can be initiated with this token.	{ "reference_number":"15102913382030662954", "card_token":"546454597415455" }
reference_number	String	Unique reference for a transaction.	

RESPONSE PARAMETER			JSON:
Field	Type	Description	
card_token	String	Unique token which represents the card that has been added.Payment transaction can be initiated with this token.	{ "Error":"", "ReferenceNumber":"15102913382030662954", "Success":true, "card_number":"4567890243749490", "card_exp_year":"2019", "card_exp_month":"09", "card_security_code":"123", "merchant_id":"VISA", "name_on_card":"XYZ" }
Error	String	If something failed	
reference_number	String	Unique reference for a transaction.	
Success	bool	True/False	
merchant_id	String	Merchant ID which represents the merchant storing the card	
card_number	String	A valid card number	

card_exp_year	String	Expiry year of the card (Format: yyyy) Example: 2019
card_exp_month	String	Expiry month of the card (Format: mm) Example: 09
card_security_code	String	The CVV number of the card.
name_on_card	String	Cardholder name

Let's have a look at comparison of Traditional vault based tokenization vs. Improved vault based tokenization approach on different parameters.

Table-1: Comparison parameters of Improved vault based tokenization w.r.to traditional

Requirement	Traditional Vault Based Tokenization	Improved Vault Based Tokenization
Performance	Slow, due to Latency	Fast. Option to deploy without latency
Scalability	Not scalable for huge customer record	Scalable because of Distributed database.
Availability & Disaster Recovery	High availability but require Sophisticated Replication.	High availability , replication is easier comparatively.
Protection of PCI data	Excellent	Excellent
Protection of PII data	Excellent	Excellent
Protection of PHI data	Excellent	Excellent
Security	Concentrated at single point.	Distributed based on Database or vault instances.
Store and protect sensitive data	Mapping of sensitive data to token is captured in single database or vault.	Mapping of sensitive data to token is captured in multiple distributed databases or vaults.

Below lists the features available with Improved Vault based Tokenization:

Table-2: Listing features of Improved Vault based Tokenization

Features	Details
Format preserving Tokenization	Complies with tokenization guidelines for token generation and allow masking of token.
Supported Token Vault Databases	Oracle & MySQL
Supported APIs	REST/JSON/Java
Logging & Monitoring	Tokenization server manages logging & monitoring for audits.
Token Formats	Random or sequential Masked: Last four, First six etc.

Sample Token vault database record:

Table-3: Token Vault or database sample records

Token	Sensitive data	Description
7896453213216520	5894589645216520	Token consisting of only Numbers with first 12 digit masked value for cardholder display experience.
7jlsoevldsuw5290	5789652346315290	Token consisting of Alphanumeric Value
7896453215896160	5796842639721560	Token consisting of Random Numbers.

Below Graph depicts the Vault lookup performance of Traditional vs. Improved vault Tokenization:

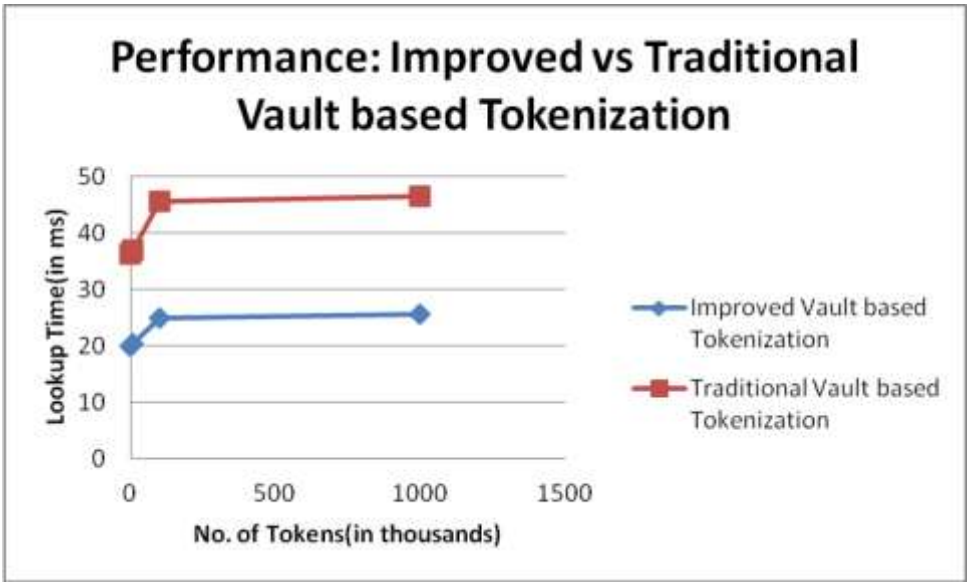

Chart-1: Performance: Improved vs. Traditional Vault based Tokenization

Table - 4: Performance statistics

Improved Vault based Tokenization		**Traditional Vault based Tokenization**	
No. of Tokens (in thousands)	Lookup Time (in ms)	No. of Tokens (in thousands)	Lookup Time (in ms)
1	20	1	36.36
10	20.3	10	36.9
100	25	100	45.5
1000	25.5	1000	46.36

In terms of percentage the overall improved performance is **55%** reduction in the latency comparatively to traditional Vault based Tokenization.

3. CONCLUSIONS

Tokenization is a vital asset when it comes to data protect in-transit. It is different from encryption which is mostly suitable for data at-rest, but at times tokenization is considered to be used in addition to encryption in payment transactions. Though tokenization is efficient way in preventing the data leak but it is unable to create secure channels or provide any kind of authentication but undoubtedly Tokenization is great for data protection. Tokenization is effectively used in software testing

like Quality Assurance, UAT database environment during the SDLC phase, because tokenization minimizes the risk of data loss. Tokenization helps to provide merchants with a system that offers highly secure way of dealing with sensitive financial data. Tokenization doesn't eliminate the need to comply with PCI DSS, but greatly helps to reduce the PCI DSS requirement by avoiding the sensitive data storage of the customer at merchant end - POS terminal or internal system.

Important principles that relates the tokenization usage and its relationship to PCI DSS compliance:

- Tokenization does the simplification for the merchant by reducing their validation efforts by reducing the number of system components requirements for PCI DSS.
- Effective implementation of tokenization application ensures that sensitive data can only be retrieved by system components falling under the scope of PCI DSS.
- Stringent security controlling & monitoring measures are required to ensure the protection of Tokenization systems and processes.
- Implementation of tokenization solutions varied in deployment models, tokenization and de-tokenization methodology, technology, and processes.
- Tokenization also helps in reducing the impact of other system components like firewall both internal as well as external, Domain Name Server(DNS) for name resolution, web routers for redirection of traffic on the security of cardholder data environment.

Tokens are generated through strong cryptographic algorithms, thereby making it difficult to re-engineered & decode. Not only the tokenization reduces the security risks significantly but also reduces the efforts required for adding tokenization in the exiting transaction processing system. Tokenization works on principle of least privilege by limiting the cardholder sensitive data exposure to only legitimate system under the compliance scope of PCI DSS. Using the mix technology of end to end encryption along with tokenization helps to preserve the data integrity and also allows financial transaction to stand a better chance against malicious attackers. Tokenization so far seen as most cost effective solution to meet the security requirement for all the entities involved in the financial transaction processing, right from the cardholder, merchants, payment network and the banks.

Furthermore, tokenization ensures enhanced security without affecting or hampering the end customer i.e. cardholder experience. For a customer there is absolutely no change in the way payment is made either on a POS terminal or an online e-commerce/ mobile commerce website/app. However, no merchant will be able to store the customer's original card number, instead of actual card number, a random number called as token is issued by cardholder's bank will be then used. In this manner the 16-digits token which masks cardholder real card number, will be dynamic in nature, ensuring that it would almost be impossible to re-engineer a token. This is why tokenization stands superior to encryption methods.

REFERENCES

[1] Shanto Roy. "Combined approach of tokenization and mining to secure and optimize big data in cloud storage." Institute of Information Technology, Jahangirnagar University, Dhaka-1342, Bangladesh. INSPEC: 17579771(2018)

[2] Sunil Dilip Jain ." Enhancing security in Tokenization using NGE for storage as a service." Sinhgad Academy Of Engineering, Kondhwa, India. INSPEC: 17411072(2017)

[3] Sandra Díaz-Santiago. " A cryptographic study of tokenization systems " Department of Computer Science, CINVESTAV-IPN, Av. IPN 2508, San Pedro Zacatenco, Mexico City 07360, Mexico. INSPEC : 16142969 (2016).

[4] Danushka Jayasinghe." Extending EMV Tokenised Payments to Offline-Environments." Tianjin, China. INSPEC: 16705386 (2016)

[5] Shafeeq Ahmad, Shreya Paul, Atma Prakash Singh." Tokenization based service model for cloud computing environment".Department of Computer Science & Engineering, AIET, Lucknow, India. INSPEC: 16620565 (2016)

[6] Tarek Kiblawi ; Ala' Khalifeh." Disruptive Innovations in Cloud Computing and Their Impact on Business and Technology". 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions). INSPEC: 7359326 (2015)

[7] Z. C Nxumalo." Towards privacy with tokenization as a service." Department of Computer Science, University of Zululand, Empangeni, South Africa. INSPEC: 15020524(2015)

[8] Ram Kumar Garg ." Developing secured biometric payments model using Tokenization." R Systems International Limited, Noida, India. INSPEC: 16072347(2015)

[9] Hassanein, Hossam & Elragal, Ahmed. (2014). "Business Intelligence in Cloud Computing: A Tokenization Approach".

[10] S. Díaz-Santiago, L. M. Rodriguez-Henriquez and D. Chakraborty, "A cryptographic study of tokenization systems," 2014 11th International Conference on Security and Cryptography (SECRYPT), Vienna, 2014, pp. 1-6.

[11] Z. C. Nxumalo, P. Tarwireyi and M. O. Adigun, "Towards privacy with tokenization as a service," 2014 IEEE 6th International Conference on Adaptive Science & Technology (ICAST), Ota, 2014, pp. 1-6.

[12] Carrol, Pat. "Tokenization: 6 Reasons The Card Industry Should Be Wary". http://www.darkreading.com/perimeter/tokenization-6-reasons-the-cardindustry-should-be-wary-/a/d-id/1316376

[13] First Data. "EMV and Encryption + Tokenization: A Layered Approach to Security". https://www.firstdata.com/downloads/thoughtleadership/EMV-Encrypt-Tokenization-WP.PDF

[14] Kuo, Li-Hsiang. "Cracking Credit Card Number Tokenization". http://pages.cs.wisc.edu/~lorderic/webpage/tokenization-crack.pdf

[15] Securosis. "Understanding and Selecting a Tokenization Solution". https://securosis.com/assets/library/reports/Securosis_Understanding_To kenization_V.1_.0_.pdf