Tokenization vs. Encryption: Options for Compliance

Version 2.0
Released: October 23, 2012

Author's Note

The content in this report was developed independently of any sponsors. It is based on material originally posted on the Securosis blog but has been enhanced, reviewed, and professionally edited. Special thanks to Chris Pepper for editing and content support.

Licensed by Intel Corporation.

About Intel:

Intel® Expressway Tokenization Broker lowers costs and dramatically simplifies PCI DSS compliance and administration for organizations across all industry types that accept, capture, store, transmit or process credit card or debit card data. Tokenization Broker replaces customers' sensitive Primary Account Number (PAN) information with secure tokens. For downstream applications that receive tokens instead of PAN data, PCI scope is either reduced or even eliminated.

Licensed by Prime Factors, Inc.

About Prime Factors:

Prime Factors is not your typical data security company. Our software and support provide specific answers you need to manage your data encryption, tokenization and key management challenges with the least amount of hassle. We call this "responsive data security" which we deliver in these ways:

- By simplifying complexity resulting in faster-time-to-deployment and lower overhead;
- Via rock solid products that have been continuously perfected based on customers' needs;
- Through support professionals who are always there to help you over the challenges that are an ongoing part of any data security software implementation.

Founded in 1981, Prime Factors leveraged our vast experience to design EncryptRIGHT from the ground up to meet PCI data encryption requirements and much more. Backed by a dedicated support team, EncryptRIGHT can be implemented by programmers and non-programmers alike to meet your data security challenges.

Copyright

This report is licensed under Creative Commons Attribution-Noncommercial-No Derivative Works 3.0.

http://creativecommons.org/licenses/by-nc-nd/3.0/us/

Table of Contents

Introduction 1

A new approach to an old problem 1

Tokenization for Compliance 1

Defining Tokenization and Encryption 2

Tokenization for Payment Data 3

Tokenization for PII 5

Tokenization for PHI 7

Conclusion 9

Who We Are 10

About the Author 10

About Securosis 10

Introduction

A new approach to an old problem

Tokenization is getting a lot of press lately because it is a disruptive security technology. Thousands of companies have replaced trusted cryptographic systems with tokenization to secure data. This is because - when applied to a common security problem such as securing credit cards - tokenization is cheaper, easier to use and more secure. Tokenization has proven its ability to protect data without impacting IT systems, and without interfering with the ability to process payments. This change-over has caused considerable confusion in the market, as many IT and security professionals incorrectly view tokenization and encryption as interchangeable technologies. For securing data at rest and in motion, encryption is the backbone technology in the IT arsenal. You simply cannot avoid encrypting to secure archives and network communications. However, in a handful of use cases, tokenization offers better security with lower cost and less management complexity.

In this research paper we will discuss appropriate uses for tokenization, and why companies look to this new approach to secure sensitive data. We get lots of questions about tokenization for sensitive data and we find many would-be customers misunderstand the technology. Some of problem is simply because tokenization is new, but more often it's because of the misleading way security technologies are marketed. Some vendors provide encryption masquerading as tokenization, while others advise using tokenization when it's not a suitable replacement for encryption. In both cases you still have sensitive information residing in databases and file servers, and you will still be reliant on encryption and key management. Tokenization can free you from many of the traditional security burdens that come with other security technologies like encryption, but the solution you select must replace - not obfuscate - sensitive data. The key goals of this paper are to provide the reader with a clear understanding of the differences between tokenization and encryption, and the types of security problems tokenization can help solve.

Tokenization for Compliance

Tokenization technology is getting huge press lately as a way to cut compliance costs, so it's logical to ask why — particularly as its value is not always clear. After all, tokenization is **not** specified by *any* data privacy regulations as a recommended technology to comply with state or federal laws. On August 12, 2011, the PCI task force studying tokenization published an "Information Supplement" called the PCI DSS Tokenization Guidelines. Commonly known as the 'Tokenization Guidance' document, it discussed dos and don'ts of using token surrogates for credit card data. While the PCI Council has **not** updated the DSS standard to include tokenization, PCI assessors are accepting it as a viable solution. Tokenization vendors are saying it helps with HIPAA, but deciding if tokenization is appropriate in healthcare requires careful analysis.

It's time to examine the practical questions regarding how tokenization is used for compliance. We will start with a brief overview of ways to use tokenization. We'll focus on use cases and value propositions, and contrast use of tokens against encryption. But before we delve into the specifics, it's worth revisiting a couple key definitions to frame our discussion:

Defining Tokenization and Encryption

Tokenization is a method of replacing sensitive data with non-sensitive placeholders: tokens. These tokens are swapped with data stored in relational databases and files. The tokens are commonly 'random' values that take the form of the original data but lack intrinsic value. A credit card number token cannot be used as a credit card to process financial transactions, except possibly in tightly controlled circumstances. Its only value is as a reference to the original value stored in the token server that created and issued the token. Note that we are not concerned with the very different 'identity tokens' used as an additional credential in multi-factor access control and authorization systems — in this paper we are only concerned with substituting tokens for data.

Encryption is a method of protecting data by scrambling it into an unreadable form. It's a systematic encoding process which is only reversible with the right key. Correctly implemented, encryption is nearly impossible to break, and the original data cannot be recovered without the key. The problem is that attackers are smart enough to go after the encryption keys, which are much easier to obtain than good encryption is to break. Anyone with access to the key and the encrypted data can recreate the original data. Tokens, in contrast, are not directly reversible.

There is a common misconception that tokenization and 'format preserving tokens' — more correctly Format Preserving Encryption (FPE) — are the same thing, but they are not. Format preserving encryption is a method of creating tokens from sensitive data. But format preserving encryption is **still encryption** — not tokenization. Format preserving encryption is a way to avoid re-coding applications and re-structuring databases to accommodate encrypted (binary) data. Both tokenization and FPE offer this advantage. But encryption *obfuscates* sensitive information, while tokenization *removes* it entirely (to another location). And you can't steal data that's not there. You don't worry about encryption keys when there is no encrypted data, or about compromise of credentials leaking access to unencrypted data.

Tokenization for Payment Data

As defined in the introduction, tokenization is the process of replacing sensitive information with tokens. The tokens are ‘random’ values which resemble the sensitive data they replace in both form and data type, but lack intrinsic value. In payment data security, tokenization is used to replace sensitive data such as bank account numbers. But tokenization’s recent surge in popularity has been specifically about replacing credit card data. The vast majority of current tokenization projects are squarely aimed at reducing the cost of achieving PCI compliance. Removing credit cards from all or part of your IT systems is good security — thieves can’t steal what’s not there. But that’s not actually why tokenization has become popular. In this use case, tokenization is popular because it saves money.

Large merchants must undergo extensive examinations of their IT security and processes to verify compliance with the Payment Card Industry Data Security Standard (PCI-DSS). Every system that transmits or stores credit card data is subject to review. Small and mid-sized merchants must go through all the same steps as large merchants **except** the compliance audit, where they are on the honor system. The list of DSS requirements is long — a substantial investment of time and money is required to create policies, secure systems, and generate the reports PCI assessors need. While the Council’s prescribed security controls are conceptually simple, in practice they demand a security review of the entire IT infrastructure.

Over the last couple decades firms have used credit card numbers to identify and reference customers, transactions, payments, and chargebacks. As the standard reference key; credit card numbers are stored in billing, order management, shipping, customer care, business intelligence, and even fraud detection systems. They are used to cross-reference data from third parties to gather intelligence on consumer buying trends. Large retail organizations typically stored credit card data in every critical business processing system. When firms began suffering data breaches they started to encrypt databases and archives, and implemented central key management systems to control access to payment data.

Has that slowed down data breaches? Not visibly. Faulty encryption deployments, SQL injection attacks, and credential hijacking continued to expose credit cards to fraud. The Payment Card Industry quickly stepped in to require a standardized set of security measures from everyone who processes and stores credit card data. The problem is that it is *incredibly* expensive to audit network, platform, application, user, and data security across all these systems — and then to document usage and security policies sufficiently to demonstrate compliance with PCI-DSS.

If credit card data is replaced with tokens, almost half the security checks no longer apply. For example, the requirement to encrypt (most or all) databases or archives goes away when you remove credit card numbers. Key management systems shrink, as they no longer need to manage keys across the entire organization. You don’t need to mask report data, rewrite applications, or reset user authorization to restrict access. Fewer controls and fewer systems to audit means less effort by PCI Assessors and internal IT staff. Tokenization drastically reduces the complexity and scope of auditing and securing operations. That doesn’t mean you don’t need to maintain a secure network, but the requirements are greatly reduced. Even for smaller merchants who can self-assess, tokenization reduces the workload and overall risk.

Tokens can be created and managed in-house, or by third party service providers — often called Tokenization as a Service. Both models support web commerce and point-of-sale environments, and integrate easily with existing systems. With in-house token platforms, you own and operate the token system, including the token database. The token server is integrated with back-end transaction systems and swaps credit cards for tokens during each transaction. The token server stores the original credit card data, but rather than stored in many locations, the numbers are in a single secure token database. This type of system is most common with very large merchants who need to keep the original card data and want to keep transaction fees to a minimum.

Third-party token services, such as those provided directly by payment processors, return a token to signify a successful payment. But the merchant only gets the token rather than the credit card. The payment processor stores the card data along with the issued token — in what's called a Token Vault in PCI parlance — for recurring payments and dispute resolution. Small and mid-sized merchants with no need to retain credit card numbers lean towards this model — they sacrifice some control and pay higher transaction fees in exchange for convenience, reduced liability, and cheaper compliance.

Deployment of token systems can still be tricky, as you need to substitute existing payment data with tokens. Token for credit card substitution must be synchronized across many systems so keys and data maintain relational integrity. Token vendors, both in-house and third party service providers, offer tools and services to perform the conversion. If you have credit card data scattered throughout your company, plan on paying a bit more for the conversion.

That said, tokenization is mostly a drop-in replacement for encryption of credit card data. It requires very little in the way of changes to systems, processes, and applications. While encryption can provide very strong security, customers and auditors prefer tokenization because it's simpler to implement, simpler to manage, and easier to audit.

Today, tokenization of payment data is driving the market. It's a proven model with thousands of merchants currently using the technology.

Tokenization for PII

As a technology, data tokenization has yet to cross the chasm — so far it has only been embraced by merchants for replacement of payment data, but our research indicates growing interest in tokenization to protect personal information. By personal information — often called Personally Identifiable Information (PII) — we mean Social Security Numbers, driver's license numbers, passport data and other sensitive personal information. Data tokenization is useful beyond simple credit card substitution — protecting other PII is its next frontier.

The problem is that thousands of major corporations have built systems around Social Security Numbers, passport identifiers, driver's license numbers, and other information that represents a person's identity. These numbers were engineered into the foundational layers of applications and business processes which organizations rely on. The ID (record number) literally ties all their systems together! Until recently, you could not open a new telephone account or get an insurance policy without supplying a Social Security Number. Not because the vendor legally needed the number, but rather because their IT systems required it to function. SSNs provide secondary benefits for business analysis, a common index for third party data services, and useful information for fraud detection. But the hard requirement to provide SSN (or driver's license number, etc.) was because application infrastructures were designed around these standard identifiers.

PII was woven into database and application functions, making it very hard to remove or replace without adverse impact on stability and performance. Every access to customer information — billing, order status, dispute resolution, and customer service — required an SSN. Even public web portals and phone systems use SSN to identify customers. Unfortunately, this both exposed sensitive information to employees with no valid reason to access customer SSNs and contributed to data leakage and fraud. Many state and local government organizations still use SSNs this way, despite the risks.

Organizations have implemented a form of tokenization — albeit unwittingly — by substituting SSN and driver's license numbers with arbitrary customer ID numbers. Social Security Numbers are then moved into secure databases and only exposed to select employees under controlled circumstances. These *ad hoc* home-grown tokenization implementations are no less tokenization than those systems offered by payment processors. A handful of organizations have taken this one step further, using third-party solutions to manage token creation, substitution, data security, and management. But there are still thousands of organizations with sensitive data in files and databases to identify (index) clients and customers. PII remains a huge potential market for off-the-shelf tokenization products.

While this model is simple, and substantially improves security, not every company uses tokenization for PII — either commercial or *ad hoc* — because they lack incentive. Most companies do not have strong motivation to protect **your** personal information. If it's lost or stolen, **you** will need to clean up the mess. There are many state regulations that require companies to protect PII and alert customers in the event of a data breach. But these laws are not adequately enforced, and provide too many loopholes, so very few companies ever pay fines. For example, most laws do not apply to breaches where data encryption was in use. So if a company encrypts network communications, *or* encrypts data archives, *or* encrypts your database, they're exempt from disclosure. The practical upshot is that companies encrypt data in *one* context — and thus escape legal penalties such as fines — while leaving it exposed in other contexts. The fact that so many data breaches continue to expose customer data clearly demonstrates the lack of effective data security.

Properly deployed, encryption is a perfectly suitable tool for protecting PII. It can be set up to protect archived data or data residing on file systems without modification to business processes. Of course you need to install encryption and key management services to protect the data, understanding this only protects data from access that circumvents applications. You can add application layer encryption to protect data in use, but this requires changing applications and databases to support this additional protection, paying the cost and accepting the performance impact.

In cases such as PII — where the original data is not needed for the vast majority of application functions — tokenizing personal information reduces the risk of loss or theft *without* impacting operations. Risk is reduced because thieves can't steal what's not there. This makes tokenization superior to encryption for security, and removes the additional overhead of managing encryption and key management systems for data at rest, archive and transport layer encryption. Nor are you worried in the event that encryption is deployed insecurely, if local administrative accounts are hijacked, or if encryption keys are compromised; in these cases the data is not exposed. Tokenization simplifies operations — PII is stored in a single database, and you don't need to install key management or encryption systems. Setup and maintenance are both reduced, as is the number of servers which require extensive security. While there is no mandate for its use, tokenization of PII is usually the best strategy because it is cheaper, faster, and more secure than alternatives.

Tokenization for PHI

Securing Personal Health Records (PHR) for healthcare providers is supposed to be the next big market for many security technologies. Security vendors market solutions for Protected Health Information (PHI) because HIPAA and HITECH impose data security and privacy requirements. If healthcare providers fails in their custodial duty to protect patient data, they face penalties — theoretically at least — motivating them to secure the data they depend on. Tokenization is one of the technologies being discussed to help secure medical information, based on its success with payment card data, but unfortunately protecting PHR is a **very** different problem.

A few firms have adopted tokenization to protect select personal information — usually a single token that represents name, address and SSN. The remainder of the data is stored in the clear as it is used for analysis. Age, medical conditions, medications, zip code, hospital, physician, heath care, insurance, and other data points should be anonymous. The problem is it has been repeatedly demonstrated that medical history and physician are enough to determine a patient's identity. A zip code and a clinical trial are enough to determine a patient's identity. An age and a zip code have a very high likelihood of determining a person's identity. Partial tokenization is ineffective for privacy because this additional data is part of the medical, billing, and treatment data which can be reverse engineered. Partial tokenization results in all of the data being at risk!

Despite this, some companies have pursued this strategy to reduce risk. It has not yet been legally tested, but organizations have a defensible position if they have substituted a person's name, address, and Social Security Number with tokens even if the rest of the data is lost or stolen. Technically they have transformed the records into an 'indecipherable' state as defined by most privacy statutes, so even if someone linked identities to the partially tokenized records these companies feel they have reduced their risk of penalties. Or at least until a court decides what 'low probability' means in terms of data privacy or HIPPA legislation, but given there are tools available on the Internet to perform this type of data 'reverse-engineering', it's not likely to hold up to scrutiny.

So while there is a lot of hype around tokenization for PHI, the model **does not work**. It's a *many-to-many* problem: we have many pieces of data which must be bundled up in different ways for many different audiences. PHI is complex and made up of hundreds of different data points. A person's medical history is a combination of personal attributes, doctor visits, complaints, medical ailments, outsourced services, doctors and hospitals which have served the patient, etc. It's an entangled set of personal, financial, and medical data. And many different groups need access to different subsets (or all) of it: doctors, hospitals, insurance providers, drug companies, clinics, health maintenance organizations, state and federal governments, and so on. And each audience needs a different slice of the data — but *must not* see the rest of the data.

The problem is knowing which data to tokenize for any given audience, and maintaining tokens for each use case. If you create tokens for someone's name and medical condition, while leaving drug information exposed, you have effectively leaked the patient's medical condition. Billing and insurance can't get their jobs done without access to the patient's *real* name, address, and Social Security Number. Tokenized medical history is useless to a doctor who needs your medical history. And if you issue the same tokens for certain pieces of information (such as name and Social Security Number) it's fairly easy for someone to guess the tokenized values from other patient information — meaning they can reverse engineer the full set of personal information. You need to issue a different token for each and every audience, and in fact for each party which requests patient data.

Can tokens work in this 'many-to-many' model? It's possible but not recommended. It's not that tokens are the issue: Tokens can take any desired format or include any type of data. The issue is in token management and mapping the tokens to many different audiences and data templates. You need a very sophisticated token management and tracking system to divide up the data, issuing and tracking different tokens for each audience. No such system exists today. Furthermore, it simply does not scale across very large databases with dozens of audiences and thousands of patients.

This is an area where encryption is superior to tokenization. In the PHI model, you encrypt different portions of personal health care data under different encryption keys. The advantage is that only those with the requisite keys can see the data. The downside is that this form of encryption also requires advanced application support to manage the different data sets to be viewed or updated by different audiences. It's still a many-to-many problem, but feasible with sophisticated key management services. However, the key management service must be very scalable to handle even a modest community of users. And since content is distributed across multiple audiences who will modify or contribute new information, record management is particularly complicated. This works better than tokenization, but still does not scale particularly well.

Masking technologies are also better suited to this use case than tokenization. Masking is a process that scrambles data, either working on an entire database or on a subset of the data. Unlike encryption, masking is not reversible; unlike tokenization, masked data is still useful for limited purposes. Masking can scramble individual data columns in different ways so that the masked value **looks** like the original — retaining its format and data type just like a token — but is no longer sensitive data. Masking also is effective for maintaining aggregate values across an entire database, enabling preservation of sum and average values within the data set, while changing all the individual data elements. Masking can be performed in such a way that it's extremely difficult to reverse engineer the original values. In some cases, masking and encryption provide a powerful combination for distribution and sharing of medical information.

Conclusion

Tokenization is an important security tool, with cost and security advantages over encryption in select use cases. With data security tools the goal is to reduce the risk of loss or exposure — benefits that encryption, tokenization, and masking all provide. Tokenization excels in this regard because you remove sensitive data. If you remove the data, you remove the risk. In some cases you can completely outsource tokenization services, removing both the sensitive data as well as your dependency on encryption, key management and other security precautions. Most users of tokenization will choose to manage their own token server, which will require encryption and key management support to safeguard sensitive data housed in the token server. But you can remove extensive security controls from the systems where tokens reside in place of sensitive data, resulting in reduced complexity and management requirements.

Today tokenization is used almost exclusively for payment card security. In this market it has demonstrated reduced audit costs and reduced complexity of security systems management - all without disruption to business operations. It's early, but we are seeing adoption for personal and health care information replacement as well. While both these use cases have hurdles to overcome before we see widespread adoption, our research indicates that the adoption trend will continue as the same tangible benefits are possible.

In general if you need to access the original data as part of a *regular* business function, encryption is a better choice. It's easier to decrypt and use a copy locally — especially with larger data sets — than to get data back from the token server. But if you don't need the original data , and merchants can use a transaction identifier rather than credit card numbers — tokenization is a great option. If you don't need sensitive data, why risk keeping it? Replace data with tokens and reduce your overall risk.

Who We Are

About the Author

Adrian Lane, Analyst and CTO

Adrian Lane is a Senior Security Strategist with 22 years of industry experience, bringing over a decade of C-level executive expertise to the Securosis team. Mr. Lane specializes in database architecture and data security. With extensive experience as a member of the vendor community (including positions at Ingres and Oracle), in addition to time as an IT customer in the CIO role, Adrian brings a business-oriented perspective to security implementations. Prior to joining Securosis, Adrian was CTO at database security firm IPLocks, Vice President of Engineering at Touchpoint, and CTO of the security and digital rights management firm Transactor/Brodia. Adrian also blogs for Dark Reading and is a regular contributor to Information Security Magazine. Mr. Lane is a Computer Science graduate of the University of California at Berkeley with post-graduate work in operating systems at Stanford University.

About Securosis

Securosis, L.L.C. is an independent research and analysis firm dedicated to thought leadership, objectivity, and transparency. Our analysts have all held executive level positions and are dedicated to providing high-value, pragmatic advisory services.

We provide services in four main areas:

- Publishing and speaking: Including independent objective white papers, webcasts, and in-person presentations.
- Strategic consulting for end users: Including product selection assistance, technology and architecture strategy, education, security management evaluation, and risk assessment.
- Strategic consulting for vendors: Including market and product analysis and strategy, technology guidance, product evaluation, and merger and acquisition assessment.
- Investor consulting: Technical due diligence including product and market evaluation integrated with deep product assessment by our research partners.

Our clients range from stealth startups to some of the best known technology vendors and end users. Clients include large financial institutions, institutional investors, mid-sized enterprises, and major security vendors.

Securosis has partnered with security testing labs to provide unique product evaluations that combine in-depth technical analysis with high-level product, architecture, and market analysis.