



POINT OF VIEW

Eliminating Card Numbers to Minimize PCI Exposure

If You've Got Nothin', You've Got Nothin' to Lose

Security professionals have to paddle fast to keep abreast of industry standards and federal regulations. Not long ago, HIPAA, Sarbanes-Oxley, and GLBA were the main compliance buzzwords. Now the Payment Card Industry (PCI) Data Security Standard (the "PCI Standard") has entered the contest. The PCI standard—which merges requirements from the Visa Cardholder Information Security Program (CISP), the MasterCard Site Data Protection (SDP) program, and other payment vendors—targets merchants and service providers that store, process, or transmit cardholder data. Besides stipulations related to network security, access control, third-party assessment, and vulnerability management, the PCI Standard requires companies to protect cardholder data and other sensitive information that they store or transmit across public networks. Non-compliance is a poor option: Fines and penalties for non-compliance can be extremely high and may include a permanent ban on participation in vendor card programs.

If your company accepts a high volume of credit cards, chances are that you have already felt the sting of PCI requirements. You have likely completed one assessment (either internally, or validated by a third party) and have come to love and hate the term "compensating control." You have struggled with process documentation and two-factor authentication. And you have stewed over the cost and complexity of implementing encryption capabilities on zero-downtime applications and legacy equipment. If wireless is deployed at your campus, you've undoubtedly lost sleep working out scenarios and solutions.

Although you can't entirely avoid card-related risk and compliance issues, you can lessen their impact by limiting storage of credit card numbers and reducing the overall scope of the PCI Standard on your organization. Simply put, thieves and hackers cannot steal card numbers that you do not have. There is no better defense against a compromise than not having anything to take in the first place. By eliminating all card numbers or only holding limited card data in a very small subset of your entire network, you can greatly narrow risk exposure and potentially reduce the impact of the PCI Standard on your organization. Doing so can help save money by limiting the range of PCI assessments and reducing requirements for encryption, key management, infrastructure maintenance, and auditing. You maintain all the functionality of payment cards, while reducing the risks and costs associated with storing payment card numbers.

Impossible you say? Not in the least.



Where it all comes together.™

As published in the March 2006 ISSA Journal



+ The Fear Factor: Losing Business Functionality

As a security consultant who has broached this idea to customers many times, I can hear the objections now. The first group to voice concern is usually Accounts Receivable. The main issue for Accounts Receivable is that they must be able to research unpaid accounts, handle chargebacks, and in some cases, uniquely identify a transaction's payment method. Even for these cases, there is an alternative to storing actual card numbers en masse. When the card number is indispensable, its use and storage can be restricted to systems dealing with the specific task at hand, for example, researching delinquent accounts. The percentage of card numbers needed for these purposes is a very small subset of the overall card transactions that most businesses perform. The benefits of this approach could be significant, and showing the folks in Accounts Receivable how the program works may help change their minds.

Marketing is next. Marketing departments LOVE the credit card number because it allows them to easily track purchases and purchasing behavior. They understand that humans are creatures of habit, who typically use one or two credit cards for all their purchases. They will argue that their tracking processes for customer data will be ruined if they can't key off the credit card number. They are mistaken.

Finally, I usually hear from the business process folks, who rely on card numbers for everything from card research, fraud detection, and loss prevention to store and front-end management. They believe they need the card number for the off chance that a point-of-sale (POS) terminal fails, the power goes out, or they have to track purchases on stolen cards. In most cases they don't need the number. Even when they do, the number can be easily retrieved in a manner that transfers risk away from the company, and back to the acquiring bank or processor (i.e., the institution that processes credit card authorizations and payments for merchants).

+ Hash—Cheap, Easy, Digestible

How do you meet these business needs while ridding your infrastructure of card numbers and trying to reduce costs when it comes to PCI compliance? You simply transform card numbers into strings of code that uniquely identify each card account without revealing the account number itself. This feat is achieved by using a one-way hashing algorithm. One-way hashing (also called fingerprinting) uses a complex mathematical formula to convert input (such as a card number) that, when run through the formula (algorithm), will always produce the same output. This allows you to use a hash as a record key, much like you would use a credit card number.

During a transaction, hashing typically is performed at the point-of-sale terminal (for in-store purchases), the e-commerce application (for online transactions), or point-of-sale software (for call-center orders). When you have to convert records in legacy systems, the hashing process is slightly more complex and time-consuming. Still, it is far more efficient and cost-effective than encryption and other methods of strong encryption. With a hashing solution, you do not have to buy any expensive software upgrades or licenses for encryption, or implement a key management process, which can consume human capital cycles. With hashing, you can use any database system and the changes you must make to the database structure are limited to the storage requirements for the credit card number, and the applications that use it.



Creating hashing functionality is very simple. While writing this article, I created a very simple hashing algorithm that you can see in the sample below. It was written in the freely available PHP scripting language and took about ten minutes to perfect (including debugging time). This algorithm needs another hour of work to be truly hardened for commercial use (error trapping, better bounds checking, and so on), but could be implemented with relative ease. The same algorithm can also be used to convert existing data sets, as PHP can integrate with almost any type of data source. The trick is to use well-known, existing hashing technologies such as MD5, SHA-1, and others to create these hashes. If you are concerned about the issues surrounding MD5 and SHA-1, try a SHA-2 implementation or something more progressive.

Pass the Salt, Please

The nature of one-way hashing is that the algorithm cannot be reversed. That means, the output from a one-way hash cannot be sent through an algorithm that reconstructs the original card number. The only way to obtain the original number would be to compute every valid card number through a hashing algorithm and then match the hashes. To ensure that two hashes of identical data yield different output (and thereby prevent hash matching), virtually every hashing algorithm includes an additional component, called a “salt.” This “salt” (also called a “seed”) is a random combination of characters that acts like a key for the algorithm. It causes the output of the hashing algorithm to be different for each particular input, as shown in the following example.

Original Value	Salt	Hash	Total Stored Value
3700000000000002	\$1\$f250cb63\$	Xsdn3YMR2V91ZOe0WNZXa/	\$1\$f250cb63\$Xsdn3YMR2V91ZOe0WNZXa/
4111111111111111	\$1\$cc741e3c\$	SiPNMkRY9.32NhV.IdmY1.	\$1\$cc741e3c\$SiPNMkRY9.32NhV.IdmY1.
5111111111111111	\$1\$0ae6e3ca\$	rjxLZbVF/p85SZtgUZWoW.	\$1\$0ae6e3ca\$rjxLZbVF/p85SZtgUZWoW.
6011000000000012	\$1\$45ea086c\$	8fq1zlqzWTaVvL9Wu/WWG1	\$1\$45ea086c\$8fq1zlqzWTaVvL9Wu/WWG1

This example illustrates a number of important points:

- I used the MD5 algorithm to generate the preceding hashes.
 - To create the salt, I computed a straight MD5 hash of the full card number. MD5 salts are twelve characters in length, and must start with \$1\$ and end with \$. This leaves eight characters to play with. I extracted the eight most significant digits (the ones at the end) from the hash to create my salt. Using this technique, every card has its own unique, reproducible salt.
 - To further mask the output, I fed the full card number into a hashing algorithm. If a hacker were to obtain a copy of the hashed database, the best he or she could do is uncover one card number at a time by using the supplied salt to calculate a hash. Think of this as encrypting each card with its own key. Depending on your hashing algorithm, this could be equivalent to 128-bits of encryption strength (with the SHA-256 algorithm). Note: This model was run multiple times to ensure accuracy. Try it yourself, and you will get the same output.
- The total stored value is the complete hash that you must store. The first 12 characters are the salt; the next 22 are the hashed card number.
- The salt must be predictably generated in order to ensure that the same input always produces the same output. I would base the salt on one of the following:
 - Characters pulled from an MD5 sum of the card number (MD5 is just a hash without a salt)



WHEN YOU MUST HAVE THE NUMBER

By its nature, one-way hashing does not allow you to “work backwards” to reconstruct a credit card number. If you are in an environment where this capability is necessary (e.g., your processor or acquiring bank does not provide access to full card numbers after the transaction has occurred), you can still implement protections to reduce your PCI exposure. To do so, move card numbers onto mid-tier or distributed systems that leverage encrypted databases in secure enclaves of your network. To ensure this approach is effective, be sure that stringent access controls (such as two-factor authentication) are in place, both at the network and user level, and that access is logged and monitored.

- Characters generated from the last four digits of the card number
 - Proprietary generation based on a combination of factors including XOR, Base64, or simple math
 - Characters pulled from an MD5 sum that is salted with a proprietary set of bits that is treated like a secret encryption key and securely stored
4. For an additional layer of protection, you can tie cards to an expiration date so that you can track whether customers renew their credit cards (e.g., for marketing purposes). To do so, add the expiration date into the hash generation process. Each time the expiration date changes, you will get a new hashed value for the customer. In addition, you may append a “secret value” to the card number to help create an even more unique salt. In this case, a hacker would have to know both the un-encrypted card number and the company secret value to validate that you have stored one card at a time.

Using this approach to create a salted hash, you can create a repeatable process by which you can store transactions and retain unique identifiers to perform the following tasks:

- Look up transactions for returns, fraudulent activity, and chargebacks
- Track transactions per card for marketing and other purposes
- Run fraud-detection algorithms

Using the Hash to Find Transactions

How do you find transactions? Well, it depends on what you are trying to do. If you have the capability to print (or barcode) the transaction identifier (e.g., an internal transaction number or the hashed card number on the receipt), you can easily look up any transaction. In addition, if your business retains paper records of transactions, you can instantly look up any other transactions for which the card was used.

To illustrate how hashes can be used to process transactions and related events, let’s examine chargebacks and fraud tracking.

Chargebacks occur when a customer claims that he or she was charged for services that were not rendered. When processors or acquiring banks return these exception files, they include the relevant card number. Simply run the number through your hashing algorithm to instantly pull any transactions you need. If you need to track these chargebacks over time, store the card number in a smaller database in an encrypted format (something that can be reversed). This should significantly reduce the overall volume of data that is subject to the PCI Standard, and allow you to focus on encrypting a smaller subset of data.

Now what if you are trying to get a card number for fraud purposes? Generally, processors allow a select number of people at a merchant to request or look up this information. A date, amount, and authorization number are generally all you need to correctly identify any transaction. Once you identify the transaction and retrieve the number from your processor, simply move it into the smaller encrypted database that meets the PCI Standard.

The Bill

Compared to other encryption schemes (asymmetric or symmetric), the beauty of a hashing solution is that it is typically less costly and complex to plan, implement, and



manage. Consider the cost of implementing hashing on legacy systems: Your only added cost is disk space and human capital to write, proof, and implement the newly created algorithm. Companies with average environments (one mainframe, one or two tandems, and multiple POS systems) should be able to implement this scheme for between \$100,000 and \$500,000. Encryption, on the other hand, will drive costs through the roof with added licensing costs, software, and internal process surrounding key management. The cost difference can be very significant!

+ Five Myths About Credit Card Numbers

When consulting with companies about overall risk reduction and PCI compliance, I frequently encounter the following concerns and misperceptions. In my experience, companies with a well-conceived and properly implemented hashing solution can perform the majority of transaction-related tasks without using the actual credit card number.

Myth #1: Our employees need the card number to do their job.

Fact: Likely, you need a unique identifier to track an individual card number through the lifecycle of a transaction. A hash of the card number will allow you to uniquely identify that card and any transaction it has been used with.

Myth #2: We cannot track credit card fraud without the card number.

Fact: You can still track fraud by using your fraud-detection algorithms to spot potentially fraudulent activity, and then identifying the common card hash that ties the transactions together. From there, you can contact your processor or acquiring bank to retrieve the original card number if needed.

Myth #3: We cannot perform chargebacks without the card number.

Fact: Your acquirer or processor will send you a chargeback file that includes the original card number. Simply run this number through the hashing algorithm to obtain its match in your records; you can then identify transactions tied to that card number.

Myth #4: Removing the card number from our environment is too difficult.

Fact: Granted, there are challenges to removing the card number from legacy systems, and re-training employees on safer ways to handle customer data, but the rewards of doing this greatly outweigh the risks. If card numbers do not exist in your environment, your chances of dealing with data theft or facing a public relations nightmare can be greatly lowered.

Myth #5: Our company does not perform e-commerce transactions; we are immune to attack.

Fact: This is nowhere near true. In fact, businesses that do not conduct e-commerce transactions are at a higher risk of exposure than those that do. This is because they generally design their workflows for ease of implementation and low cost, rather than security. The perception may be that criminals target online stores to steal credit card data, but the reality is that traditional retailers may be more popular targets. That's because sellers in offline transactions usually swipe the actual credit card.¹ When the card is swiped at some retailers, the full track data is captured and stored to disk. If this data were stolen, an attacker could then re-encode blank magnetic stripes to act like real credit



THE ILLUSION OF IMMUNITY

Even if PCI compliance were not an issue, risk management is a compelling driver for reducing credit card numbers in your environment. Many companies are under the false impression that they are immune to fraud and its consequences. These merchants believe that there is no risk in storing credit card information on their mainframes and in other places. "Oh, it's secure. We use access control and we have a firewall too." Many think that the risk is transferred because they bear no direct financial impact. Let's clear the slate on this thinking.

Everybody loses when credit card information is compromised. Data loss may directly affect the perceived value of card brands (such as Visa and MasterCard) and the issuing banks of the stolen card accounts. Depending on circumstances, these companies must absorb the actual dollar value of the loss as well as other damages, including loss of brand equity and consumer confidence.

A merchant can also experience significant and far-reaching damage if its network, internal systems, or other data repositories are breached and credit card information is stolen. You may be subject to fines and lawsuits for failure to protect credit card data. Your credit card vendor may drop you from its program. Perhaps most important, your brand loses value. With the fear of identity theft growing,² consumers will shop at another brand when they learn that a merchant has exposed their credit card number to risk. In a society that increasingly conducts cashless transactions, lack of trust in credit (and debit) card security can profoundly impact not only the number of shoppers that you attract, but also the buying behavior of those consumers, who tend to spend more when they use their card instead of cash.³

cards. Even if this data is not stored, the brick-and-mortar types are still juicy targets because simple security mechanisms like firewalls, access controls, and encryption come at a price that is rarely seen in these environments.

+ Summary

As the number of data security regulations and standards climbs, security professionals are becoming increasingly creative in order to quickly and cost-effectively meet compliance requirements. One tactic for reducing compliance costs and complexity is to minimize exposure. One-way hashing minimizes exposure to credit card risk by allowing companies to eliminate or reduce the number of credit cards they must protect. It's fast, affordable, and can be used in legacy applications. In spite of popular misconceptions, hashes can be used for many business processes, including tracking fraud and consumer behavior, following transaction lifecycles, and settling returns and chargebacks. The secret to obtaining appropriate protection and full functionality from a hashing solution is to apply a thorough assessment, sound policies, and technological expertise. If your organization lacks the time or resources to meet these requirements, consider contracting with a well-established consultant that has a proven track record in these areas.

+ Learn More

For more information about VeriSign® Global Security Consulting, please call 650-426-5310, email enterprise_security@verisign.com, or visit us at www.Verisign.com.

+ About the Author

Branden R. Williams, CISSP, CCSA, CCSE, has nearly a decade of experience in information security. As a member of the VeriSign Global Security Consulting team, he has led application security assessments for clients in the financial, retail, service provision, and industrial sectors. He was recently appointed National Practice Lead for PCI compliance.

+ About VeriSign

VeriSign, Inc. (Nasdaq: VRSN) operates intelligent infrastructure services that enable and protect billions of interactions everyday across the world's voice and data networks. Additional news and information about the company is available at www.verisign.com.

Visit us at www.Verisign.com for more information.

- 1 Joris Evers, "Putting the Squeeze on Credit Card Fraud," News.com, September 9, 2005, http://news.com.com/2102-7349_3-5856625.html, (Accessed September 10, 2005)
- 2 Identity Theft Resource Center, "The Growing Fear of Identity Theft," http://www.idtheftcenter.org/html/growing_fear.htm
- 3 Gregory Papajohn, MasterCard International, "Debit Grows Its Share of Purchases Under \$100," May 06, 2004, <http://www.mastercardintl.com/cgi-bin/newsroom.cgi?id=871&category=keyword&keyword=debit%20cards>, (Accessed Oct 10, 2005)

©2006 VeriSign, Inc. All rights reserved. VeriSign, the VeriSign logo, "Where it all comes together," and other trademarks, service marks, and designs are registered or unregistered trademarks of VeriSign and its subsidiaries in the United States and in foreign countries. All other trademarks are the property of their respective owners.