The Art of the Compensating Control

By Branden R. Williams – ISSA member, North Texas, USA chapter

This article describes compensating controls: what they are, how to create them, what situations may or may not be appropriate, and what to avoid.

Few payment security professionals can find a hotter topic than compensating controls. They always look like this mythical accelerator to compliance used to push PCI compliance initiatives through completion at a minimal cost to your company with little or no effort.

Compensating controls are challenging. They often require a risk-based approach that can vary greatly from one qualified security assessor (QSA) to another;[1] There is no guarantee a compensating control that works today will work one year from now, and the evolution of the standard itself could render a previous control invalid.

My goal for this article is to paint a compensating control mural. After reading this article, you should know how to create a compensating control, what situations may or may not be appropriate for compensating controls, and what land mines you must avoid as you lean on these controls to achieve compliance with the Payment Card Industry Data Security Standard (PCI DSS).[2]

What a compensating control is

In the early years of PCI DSS (and even my experience under the CISP program), the term compensating control was used to describe everything from a legitimate work-around for a security challenge to a shortcut to compliance. If you are considering a compensating control, you must perform a risk analysis and have a legitimate technological or documented business constraint before you even go to the next step. We will see more of the documented business constraints coming our way for review based on the current economic situation. Just remember the word ***legitimate*** and the phrase ***perform a risk analysis*** before proceeding to the next step. Bob's being on vacation is not a legitimate constraint, and an armchair review of the gap and potential control is not a risk analysis. QSAs should ask for documentation during a compliance review, and having it ready to go will make sure you are efficiently using their time. If they do not, you can bet that your assessment will not be thorough.

Every compensating control must meet four criteria before it can be considered for validity:

1. Meet the intent and rigor of the original PCI DSS requirement
2. Provide a similar level of defense as the original PCI DSS requirement
3. Be "above and beyond" other PCI DSS requirements (not simply in compliance with other PCI DSS requirements)

1 Why? Because we are not provided a common risk model to use.

2 Please visit http://www.pcisecuritystandards.org.

4. Be commensurate with the additional risk imposed by not adhering to the PCI DSS requirement[3]

For an example of a completed compensating control, review the Appendix C of PCI Version 1.2.

An example of a valid control might be using extra logs for the *su* command in UNIX to track actions executed under a shared root password. In rare cases, a system may not be able to use something like *sudo* to prevent shared administrator passwords from being used.[4]

What a compensating control is not

Compensating controls are not a short cut to compliance. In reality, most compensating controls are actually harder to do and cost more money in the long run than actually fixing or addressing the original issue or vulnerability.

Imagine walking into a meeting with a customer that has an open, flat network, with no encryption anywhere to be found (including on their wireless network which is not segmented either).[5] Now imagine someone in internal audit telling you not to worry because they would just get some compensating controls. Finally, imagine they tell you this in the same voice and tone as if they were going down to the local drug store to pick up a case of compensating controls on aisle five.

Compensating controls were never meant to be a permanent solution for a compliance gap. Encryption requirements on large systems were made unreasonable early in this decade. Not only was there limited availability of commercial off-the-shelf software, but it was prohibitively expensive to implement. For Requirement 3.4 (Render PAN, at minimum, unreadable anywhere it is stored), card brands (largely Visa at the time) were quick to point out that compensating controls could be implemented for this requirement, one of those being strong access controls on large systems.

For mainframes, assessors would typically do a cursory walk through the controls and continue to recommend an encryption solution at some point for those systems. At one point, compensating controls were deemed to have a life span, meaning that the lack of encryption on a mainframe would only be accepted for a certain period of time. After that, companies would need to put encryption strategies in place.

Compensating control life spans never materialized. Compensating controls can be used for nearly every single requirement in the DSS – the most notable exception being permissible storage of sensitive authentication data after authorization. There are many requirements that commonly show up on compensating control worksheets, Requirement 3.4 being one of them.

Even with no defined life span, compensating controls are not an eternal free pass. Part of the process during every annual assessment is to review all compensating controls to ensure that they meet the four requirements as currently defined by the PCI Security Standards Council,[6] the original business or technological constraint still exists, and it proves to be effective in the current security threat landscape. If certain types of attacks are on the rise and a certain compensating control is not effective in resisting those attacks, it may not be considered OK on your next assessment.

To further cloud the situation, it is up to the QSA performing the assessment to decide to accept the control initially, but the acquiring bank (for merchants) has the final say. Substantial documentation and an open channel of communication to your acquirer is essential to ensure money is not wasted putting together controls that ultimately do not pass muster.

Don't get discouraged, though! Compensating controls are still a viable path to compliance even considering the list of reasons why you may not want to use them.

I would not be a true security professional if I did not have a fun story or two based on my experiences coaching companies or individuals to better security. No names will be used, and I am going to change enough details to protect those who were most likely being forced to try the old "push back on the auditor" routine. I hope you enjoy reading them as much as I enjoyed listening to them.

The funniest controls that you did not design

Some of my most cherished stories and experiences come from customers and vendors that had the right intentions but never seemed to follow the basic doctrines listed above on how good compensating controls are made.[7]

During my career I did some IT auditing for a bank that was owned by my employer. I know the drill of responding to auditor findings. They usually start with a meeting bringing all the key stakeholders together to mull over a spreadsheet listing all the findings. Findings are separated out in the "To Fix" pile, and the "To Push Back" pile, each item being assigned to an expert to push back on the auditors. "We don't need that control because of a control over here" or "This gap does not apply to our environment" are common phrases heard in these meetings. Eventually, a happy (potentially unhappy) medium is established, and the audit is closed out.

The same process is often applied to PCI, and the compensating control Cha Cha commences.

Before I poke fun at the following examples, please understand that I am only illustrating a point. At no time were these

3 As described in the PCI Security Standards Glossary: https://www.pcisecuritystandards.org/pdfs/pci_dss_glossary.pdf.

4 If you are reading this and saying, "HEY! We CAN just use shared passwords!" please grow up. Nearly every system has the ability to use something like sudo which is free, or a commercial variant.

5 While it is not a requirement to segment your network, it does make compliance easier. Usually in this situation, I find a legacy system that cannot be patched or upgraded, but now becomes in scope. Then the conversation about compensating controls starts.

6 Remember, the requirements can change between versions of the standard.

7 By the way, if you read this and think, 'Hey! He is talking about ME!?', I'm not. I promise.

suggestions made by people who didn't understand both the requirement and the capabilities of the technology in question. These people were professionals, and based on their credentials and experience, they should have known better.

Encryption has always been a hotly debated topic from the early "Just Do It"® message that was pounded into our heads, to the cooler-headed "Slow down, it's a mainframe" axiom that we live by today. My favorite failed compensating control for Requirement 3.4 comes from a vendor that called me late one afternoon. They brought in their product team and tried to convince me that RAID-5 was essentially an equivalent to encryption. Their argument stated you could not take any one drive and reconstruct useful data that could be considered compromise worthy, thus their product should be considered valid to sell to companies as encryption.

So if one drive (probably damaged) falls off of a truck during transport, the technology does prevent someone from reconstructing all the data that was on that system. If the system was large enough, chances are that the data on the drive may not provide any use to nefarious individuals either. But that's not really the goal of the requirement, is it? Physical theft prevention is covered in other areas of the standard. The point of the requirement is to render the data unreadable anywhere that it is stored. RAID may render the data unreadable on one physical drive, but it does not render it unreadable in any other circumstance. A simple compromise of one area of the system could lead to the access and theft of massive amounts of unencrypted data.

Speaking of encryption, disk-only encryption inside data centers is not very useful either, unless additional user credentials are tied to the decryption process. Another favorite was a vendor that offered PCI compliance through an encryption appliance that was completely transparent to the operating system. So basically, you were only protecting the data as it sat on disk, in a secured facility, with gates, cameras, and Buck, the not-so-friendly security guard that looks like a hiring manager gave a night shift and a taser to the ex-bouncer of a dance club. If applications sat on disk drives housed in the unlocked part of a post office, then I could see the value here. Until then, the solution only focuses on the physical media and nothing else.

Encryption is really not the big problem with Requirement 3; key management is. Once companies figure out that encryption technologies are available for their platforms, they realize that key generation and management is a whole different problem. One vendor who apparently thought I had already checked out for the weekend make a case for using the COBOL Random Number Generator (RNG) to spit out sixteen digits (technically 128 bits of data) to use as an encryption key.

Yes, you are trying to be random and you will end up with a 128-bit key. Anyone with a basic knowledge of encryption will quickly find the problem with that approach. Not that COBOL's RNG is less than R, but that you have eliminated a giant section of possible key space! A 128-bit key generated in that manner is the equivalent of (approx.) 53 bits of encryption, thus making it computationally feasible to brute force that key.[8]

How to create a good compensating control

We have spent quite a bit of time setting this section up. We talked about what compensating controls are, what they are not, and some of the best mis-guided attempts to create them. Before discussing the examples, please remember that they should be used for illustrative purposes only. I have over simplified the scenarios for brevity, and things are rarely this simple in the corporate world. Ultimately, compensating controls must be approved first by a QSA, or barring that, your acquiring bank. I know I do not like it when someone brings an article about PCI to an interview during an assessment, so please don't do that with this one. Now let's walk through a couple of examples of how one might create a good compensating control.

Here is a common compensating control that my team defined and implemented at a customer. A Level 1 brick and mortar retailer with 2,500 stores has some systems in their stores that do not process cardholder data. These systems are a high risk to this customer's cardholder environment because they may access both the Internet through a local firewall and the corporate intranet and webmail system, and users log-in to those machines with the default administrator account. Store managers and retail operations claim that the systems are required for day-to-day business because each store is empowered to customize their operations to better fit the local market. The corporation believes this drives innovation and helps them maintain a competitive edge over their peers. (See Figure 1)

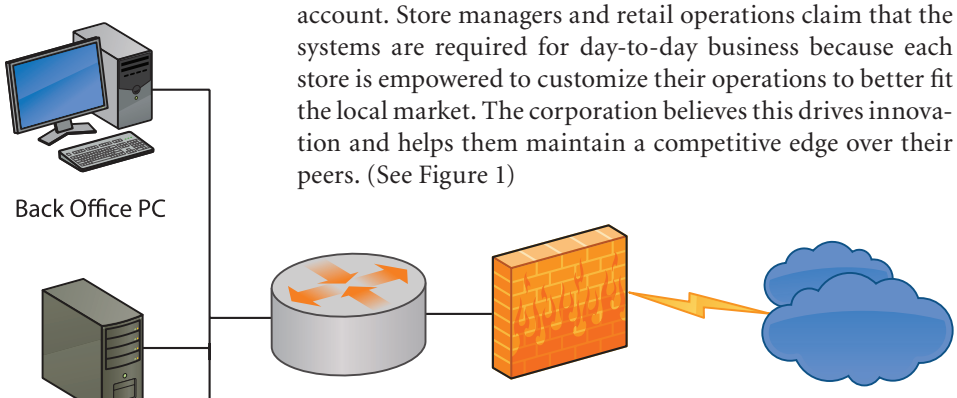

Figure 1

If the retailer chooses not to segment the network, all of the systems in the store are now in scope, and they must meet all of the applicable requirements of the PCI DSS. Doing this will add significant expense to the IT infrastructure, and will probably force a call center to be staffed up in order to man-

8 50 computers could do it in less than one year.

age the volume of calls that will come in for things like password maintenance.

What do you do? Do you crush the retailers aspirations to innovate by telling them they must deploy active directory to these machines, lock them down Department-of-Defense-tight, and staff a call center? I suppose you could. But, if you made that recommendation you missed something important – understanding the business and limiting the impact that your compliance recommendations make. Instead, consider this compensating control:

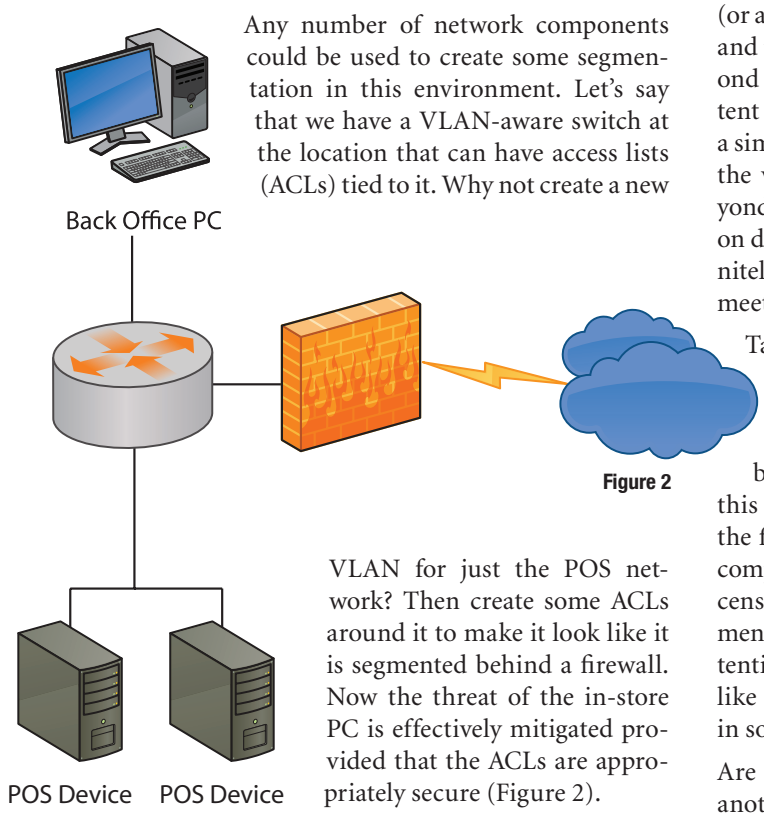

Figure 2

Any number of network components could be used to create some segmentation in this environment. Let's say that we have a VLAN-aware switch at the location that can have access lists (ACLs) tied to it. Why not create a new VLAN for just the POS network? Then create some ACLs around it to make it look like it is segmented behind a firewall. Now the threat of the in-store PC is effectively mitigated provided that the ACLs are appropriately secure (Figure 2).

> Wait a second, ACLs? Those are not supposed to be used for compliance with PCI! They most certainly can be used for compliance. Requirement 1.3.6 only refers to external connections, not internal connections. Using ACLs internally is perfectly acceptable. If you want an extra boost in security, use a reflexive access list (RACL) which will basically look and feel like a stateful inspection firewall.

"But Branden," you say, "my store networks are different in every store! I cannot just slap something in there like that and expect it to work globally!" If this is the case, I bet your store support group is overloaded with break-fix calls. Maybe this could be an opportunity to shore this up and get consistent footprints in each store?

Barring that, how about this twist?

Let's say that you are running a Windows XP variant as the operating system powering your POS. You are already required to put some kind of antivirus and malware-removal tools on there. Most of those come with software-based firewalls that could be administered remotely. Deploying firewall capabilities to the POS itself could be viewed as appropriate segmentation depending on the policy attached to that firewall. It is neither a transparent solution, nor is it very pretty, but it works.

The first solution above is really less of a compensating control and more of a way to reduce the scope of PCI. The best thing you can do for your company is reduce the scope of PCI (or any compliance initiative) to the bare minimum required, and then manage that subset of your infrastructure. The second truly is a compensating control. It meets the original intent and rigor of the original PCI requirements and provides a similar level of defense as the original requirements (reduce the vulnerability to payment systems), goes above and beyond the base requirements of PCI (firewalls are not required on devices that do not leave the premises), and it is most definitely commensurate with the additional risk imposed by not meeting the original requirement.

Take a closer look at those two suggestions. The first may be "free" to your company depending on what is already in place! You will need to adjust business process and prepare your IT community to deal with the change, but you may not need to spend any hard dollars rolling this solution out (unless your equipment cannot do this in the first place). The second suggestion, which is actually the compensating control, requires capital outlay for software licensing and training or consulting to build out the environment. Upon rollout, things will break that will result in potential losses to the business. I've seen retailers push changes like this to large environments, and every single one results in some kind of error.

Are you starting to get the hang of this thing? How about another example?

A service provider has a large mid-tier UNIX[9] installation that runs critical areas of the payment process, including long-term data storage. For various reasons, encrypting the data is not an option on these machines. How do we make this service provider compliant with PCI Requirement 3.4?

This is a real world example that comes up frequently. Encryption implementations have come a long way since early in this decade. The words "my platform does not have a solution for encryption" is no longer valid for platforms that can comply with PCI. When I present the following control to customers, it is shocking how fast they find a way to encrypt their data.

Most mid-tier UNIX operating systems have the ability to switch from Discretionary Access Control (DAC) to Mandatory Access Control (MAC). MAC will cause that mid-tier UNIX machine to act like a mainframe using RACF/ACF2, and managing those controls is now a massive chore for whoever is charged with it. Converting the appropriate systems

9 For example, Solaris or AIX.

to MAC, and potentially adding some segmentation, could effectively render cardholder data unreadable and meet PCI Requirement 3.4.

Things are never that easy. Security professionals inside companies love the idea of converting to MAC as it allows us to have more granular control over the systems and their data. Practical ones know that converting an existing system requires so much effort that the costs outweigh the benefits. This is a perfect example of how a compensating control might look good on paper (it is only three words when you use the acronym! "Convert to MAC!"), but in reality would be much easier to just meet the implied requirement to encrypt that data.

One more example, and then it is time for you to get creative!

A medium-sized retailer with less than 500 stores is struggling with requirement 10.2.1 to log "all individual accesses to cardholder data." All of their data is stored in a large DB2 database that runs on a mainframe. They run massive batch processes at regular intervals, and their space constraints prevent logging every single access to a row. Do you tell them to go back to their board for a CapEx request to buy lots and lots of drive space to store logs?

Before we proceed, consider the intent of the requirement. Reliable logs are valuable in investigating a breach quickly. Without them, it may take forensic examiners days, or even weeks, to determine the source of a breach. Once the source has been identified and analyzed, forensic companies must attempt to determine how many card numbers may have been exposed. If there are no logs, the assumption is that everything could be exposed, meaning that fines will add up pretty quickly.

The idea is not necessarily to make a log record that includes every single card number that is accessed, but to be able to identify which cards are accessed through the data contained in the logs. If we were to log the actual query performed against the database during a batch process, with knowledge of the date and time that the query was run and exactly what that query will do, we should then be able to determine, with reasonable certainty, which cards were accessed. Common batch processes run on a daily basis, usually using the data from the previous day to produce its output. If we must determine what could have been exposed from January 1 to January 8, we could look at the data that would have been accessed by that batch process during those days.

Logging the query, and all the other elements required by 10.3 about that action, would generate a reasonably accurate list of records that would use a fraction of the drive space required by creating an entry that has every single record exposed.

Go forth and compensate!

What a pretty mural we have painted over the last several pages! Good compensating controls are the result of a marriage between art and science. We have discussed what compensating controls are, what they are not, some funny examples of how to go wrong, and three solid scenarios from which we created good controls.

Compensating controls are not the golden parachute of compliance initiatives. They require work to build effective ones that will pass the scrutiny of both a QSA and an acquiring bank (or card brand). Rarely do they yield lower cost and effort than simply meeting the original requirement. PCI DSS is based on many good (not best) standards of practice for security, and should be viewed as a baseline by which to operate, not a high water mark by which you aspire to be one day. Compensating controls may help you lower the bar of compliance in the short term, but remember, only you can prevent a security breach.

About the Author

Branden R. Williams, CISSP, CISM, is the Director of the PCI Consulting Practice at VeriSign and regularly assists top global retailers, financial institutions, and multinationals with their information security initiatives. He was recently named Payment Security Professional of the Year for 2008 by the Society of Payment Security Professionals. Read his blog at http://blogs.verisign.com/securityconvergence, or reach him directly at bwilliams@verisign.com or http://www.brandenwilliams.com.